IMAGE SUPER-RESOLUTION USING ADAPTIVE 2-D GAUSSIAN BASIS FUNCTION INTERPOLATION

THESIS

Terence D. Hunt, Second Lieutenant, USAF

AFIT/GE/ENG/04-15

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/ENG/04-15

IMAGE SUPER-RESOLUTION USING ADAPTIVE 2-D GAUSSIAN BASIS FUNCTION INTERPOLATION

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Terence D. Hunt, B.S.E.E.

Second Lieutenant, USAF

March 2004

AFIT/GE/ENG/04-15

IMAGE SUPER-RESOLUTION USING ADAPTIVE 2-D GAUSSIAN BASIS FUNCTION INTERPOLATION

Terence D. Hunt, B.S.E.E.
Second Lieutenant, USAF

Approved:

Dr. Steven R. Gustafson (Chairman)	4 Mar 04 Date
Dr. Kenneth W.Bauer (Member)	5 MAR 04 Date
Maj Matthew E. Goda, Ph.D. (Member)	5 MAR 04 Date

Acknowledgements

I would like to thank my committee for their suggestions, especially Dr Gustafson for all the time spent focusing my ideas. Also thanks to the computer lab crew and my friends for all the encouragement.

Terence D. Hunt

Table of Contents

Page

List of Figures

Figure Page

Figure Page

AFIT/GE/ENG/04-15

ABSTRACT

Digital image interpolation using Gaussian radial basis functions has been implemented by several investigators, and promising results have been obtained; however, determining the basis function variance has been problematic. Here, adaptive Gaussian basis functions fit the mean vector and covariance matrix of a non-radial Gaussian function to each pixel and its neighbors, which enables edges and other image characteristics to be more effectively represented. The interpolation is constrained to reproduce the original image mean gray level, and the mean basis function variance is determined using the expected image smoothness for the increased resolution. Test outputs from the resulting Adaptive Gaussian Interpolation algorithm are presented and compared with classical interpolation techniques.

IMAGE SUPER-RESOLUTION USING ADAPTIVE 2-D GAUSSIAN BASIS FUNCTION INTERPOLATION

I. Introduction

1.1 Problem Statement

Digital images are often not recorded with a high enough resolution to enable enlargement of a portion of the image on the order of 20x20 pixels without severe distortion. Computers make it easy to zoom in on any portion of an image; however the usual result exhibits blocking phenomena resulting from the digital nature of the image (see Figure 1). This blocking is the result of standard digital interpolation, indicating that the sampling rate of the image is too low for the desired level of magnification, and it can make enlarged portions unrecognizable. This artifact clearly limits the usefulness of digital images, increasing the demand for costly optical lenses and high resolution digital sensors. [13]

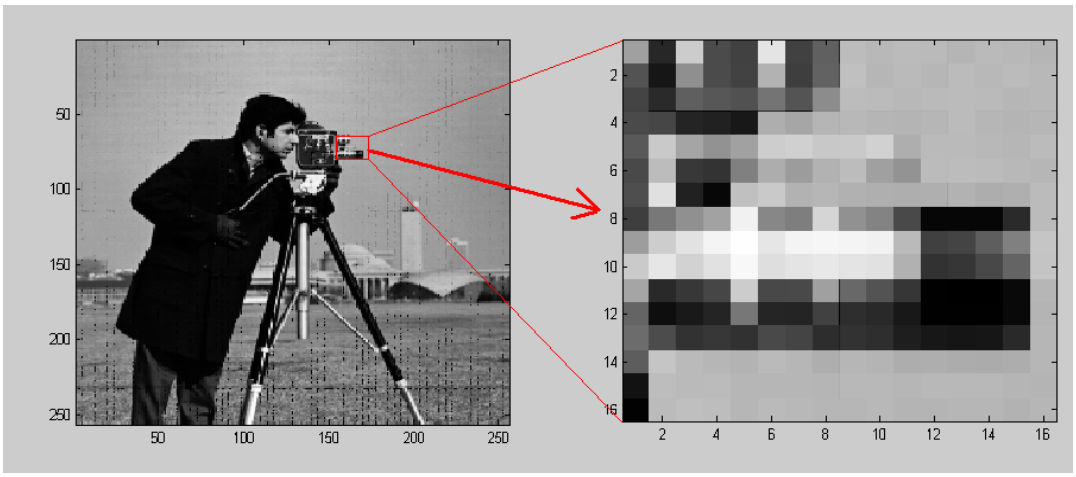


Figure 1. Example of blocking phenomena that results from the magnification of a digital image

A second problem facing digital imaging is that current technology is fast approaching the limits in pixel density of a CCD array. The optimally limited pixel size was calculated to be about 40 square microns only one year ago [12]; however, current sensor technology has already surpassed this level. Recent advances have been able to overcome what was thought to be the pixel size limitation by intensifying the light focused on the pixels. Currently, CCD arrays with pixels as small as 6.8 microns are in use according to Kodak [3]. The maximum resolution of a camera is determined by a combination of the size and quality of the camera optics, the CCD array size and the number of sensors per unit area in the array. Even assuming a perfect lens, the resolution is still limited by the digital sensors. Sensors so small that not enough photons enter them are subject to shot noise and thus do not give consistent outputs. Simply enlarging the size of the array greatly increases the cost, in addition to the resulting capacitance issues that make this an impractical alternative [12]. The capacitance issue will likely also be overcome; however, as technology advances, the cost of these devices continues to climb, motivating the development of an alternative means of resolution enhancement.

Classical interpolation commonly includes both bilinear and bicubic techniques. Bilinear interpolation is a very fast and reasonably accurate method; however, it typically results in a blurred image in which sharp edges are not maintained. Bicubic interpolation attains significant improvement over bilinear interpolation; however, it also suffers from blurring effects, although to a lesser extent (See Figure 2) [2 ,10].

The random distribution of light entering a photosensitive cell in a CCD array can be described using a two dimensional Gaussian or normal distribution in the spatial domain [4]. Thus, Gaussian radial basis function interpolation shows significant

improvement over both bilinear and bicubic interpolation using both mean square error in gray level and subjective human evaluation as metrics (see Figure 2e). Gaussian radial basis functions are flexible because their variance parameters allow a range of image smoothness. Past research has shown promising results using Gaussian basis functions for digital image interpolation [5], and this research is extended here.

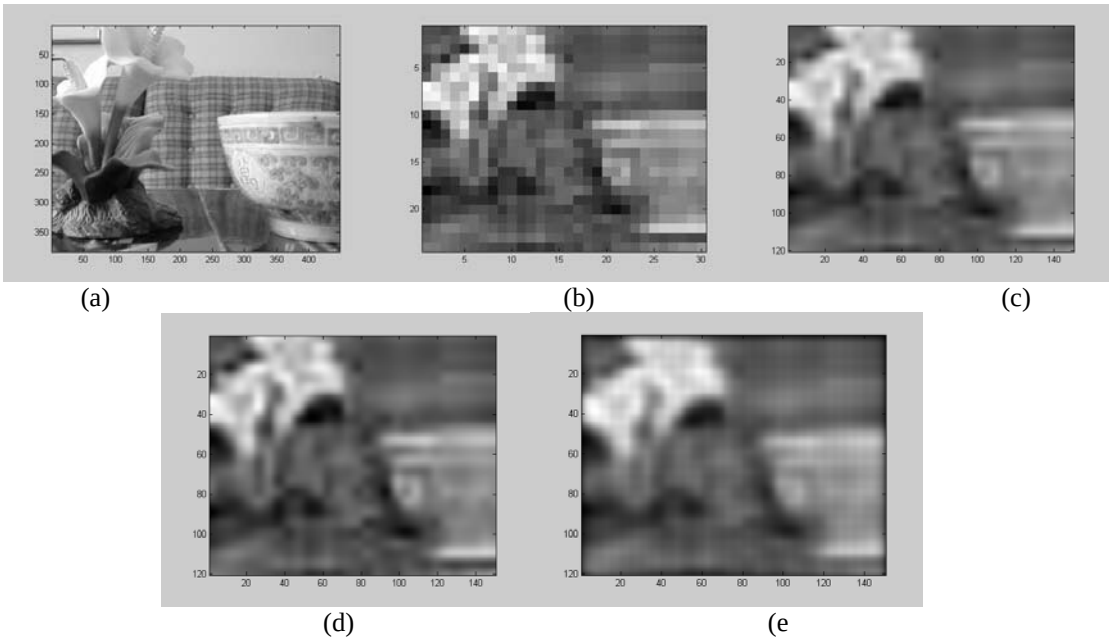


Figure 2. Image interpolation examples. (a) Original image from a digital camera, (b) Original image from greater distance resulting in degraded resolution, (c) Bilinear interpolation of degraded image, (d) Bicubic interpolation of degraded image, (e) Gaussian radial basis function interpolation of degraded image using variance 2.75.

1.2 Scope

The Adaptive Gaussian Interpolation (AGI) algorithm developed here can be applied to any digital image. AGI does not add any new information to the image, it simply transforms the data that is present to make it easier to understand or interpret, and it can improve the ability of a person to recognize an object in a low resolution situation or increase the ability of a machine to recognize a pattern that may be obscured by

blocking artifacts. This research shows that the AGI technique can improve the resolution of any image with less error than several commonly used interpolation methods.

1.3 Organization

Chapter two focuses on research already accomplished in the area of interpolation and super-resolution, and offers some background information regarding 2-D Gaussian fitting. This chapter also discusses how adaptive functions are constructed and why they typically outperform non-adaptive functions. Advantages and drawbacks of adaptive functions in general are also discussed.

Chapter three explains how Adaptive Gaussian Interpolation works in a step-by-step recipe format, with explanations for each operation and possible alternatives for each step. Each step is illustrated using figures, diagrams, and equations.

Chapter four shows results and AGI outputs. Primarily, this chapter compares image outputs from AGI with other interpolation methods and describes relative advantages and disadvantages. Both objective and subjective results are shown and analyzed. Frequency domain analysis and edge detection performance of the interpolated images are considered, as are examples of how each variable affects output.

Chapter five discusses potential future and follow on work. Such work includes using alternate basis functions such as Gabor or wavelet functions and adapting the AGI method to 3-D stereo-scopic imaging.

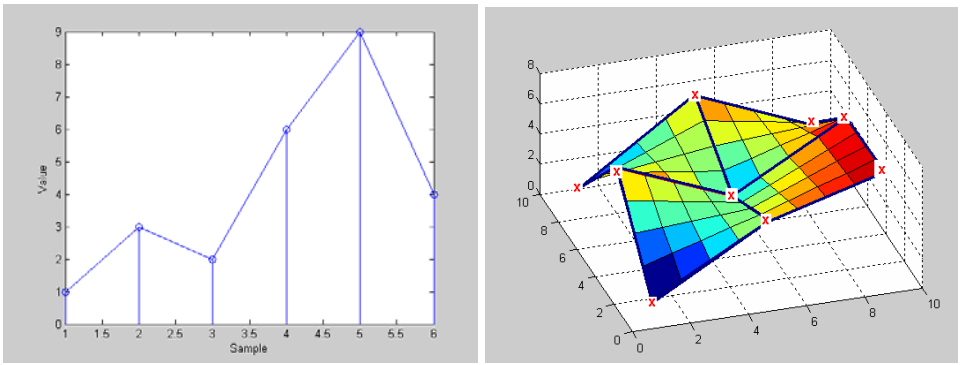
II. Background

Interpolation is constantly done by the human brain; for example if a light flashes fast enough, it is interpreted as a constant source. Similarly, if the pixels in a digital image are small enough they are interpreted as a smooth surface. However, like a light that flashed slowly enough to be distracting, a low resolution image can look “blocky” and be difficult to interpret. This chapter focuses on classical interpolation techniques on which AGI is based. The characteristics of AGI are also discussed.

2.1 Classical Interpolation

Classical interpolation techniques typically involve fitting an equation to a set of data. Two types of classical interpolation are used as benchmarks here: Linear and Cubic.

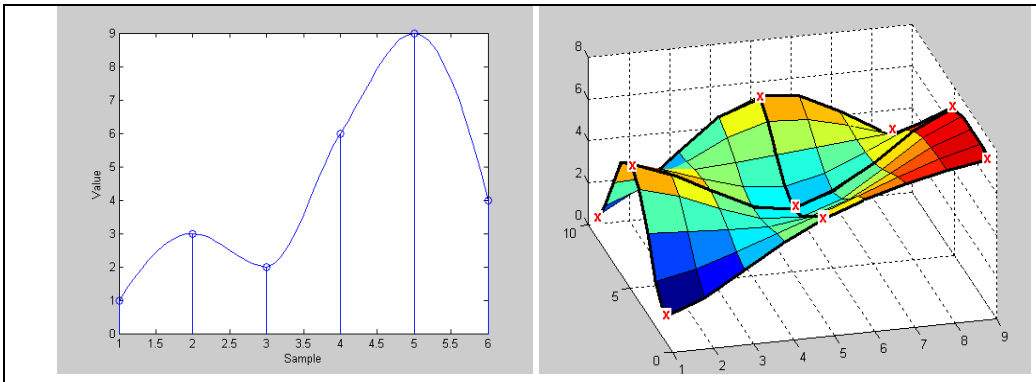
Linear interpolation uses straight lines to connect data points. This method can be adapted to 2-D which images, where it is called bilinear interpolation. Linear and bilinear interpolation techniques are illustrated in Figure 3.



(a) (b)

Figure 3 Examples of linear interpolation. (a) 2-D linear interpolation with the sample number indicated on the x-axis and the sample value on the y-axis. The circles represent the real data points. (b) An example of a 2-D bilinear interpolation. Red ‘X’s indicate real data points.

Linear interpolation fits points to first order equations, and this procedure can easily be extended by fitting higher order equations to more points to obtain more accurate representations. Thus cubic interpolation fits four data points to a third order equation, and bicubic is the 2-D extension of cubic interpolation. 1-D cubic and 2-D bicubic examples are shown in Figure 4.



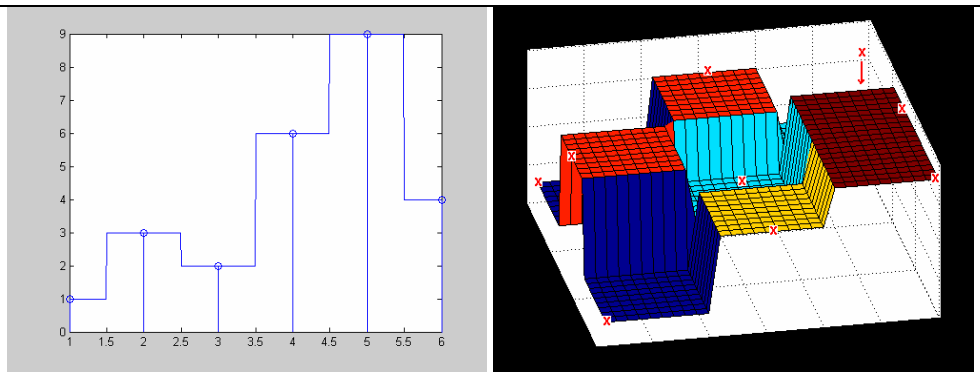
(a) (b)

Figure 4 Examples of cubic interpolation. (a) An example of 1-D cubic interpolation of the same data used in Figure 3a, (b) Bicubic interpolation of the same data used in Figure 3b. Note the increased smoothness over linear interpolation.

Nearest neighbor interpolation is a very fast technique in which it is assumed that the point to be interpolated has the same value as the nearest sampled value. This technique results in a very digital-looking output which is illustrated in Figure 5 where 1-D and 2-D nearest neighbor interpolation is demonstrated.

2.2 Basis Function Interpolation

Basis function interpolation uses predetermined functions to represent data points. These functions are fit to the data by variables that change to achieve the required amplitudes. Studies have found that basis function interpolation results in more accurate



(a) (b)

Figure 5 Example of nearest neighbor interpolation. (a) The 1-D case using the same data as in Figures 3a and 4a, (b) The 2-D case using the same data as in Figures 3b and 4b with the red X's marking the real data points.

interpolation; however, an increase in computation time is also characteristic [14]. A method using 2-D radial Gaussians has been found to accurately represent data points [5,7]. These Gaussians can be considered one dimensional Gaussians spun around their axis, which greatly reduces their complexity; see equation 2.1 for the two dimensional Gaussian equation. Also, Fourier transforms use infinite basis functions to represent data, while wavelets use more complex waveforms to show both space and frequency information. More information on Fourier and wavelet transforms can be found in [11] and [1].

2.3 Adaptive Basis Functions

The processing power of modern computers allows much more complex interpolation techniques to be executed in reasonable times than in the past. However, radial Gaussian basis functions have a limited adaptive capability. In previous research, the magnitude of the Gaussian is set to equal the pixel gray level it represents, or some scaling of that level; thus the function adapts to each pixel. However, the variance

parameter is typically a fixed number. The data surrounding the pixel being interpolated may be used to calculate an average variance, which effectively adapts the function to the local data it represents. [13, 8] Using this concept, basis functions with more variables have more adaptive capacity and theoretically yield more accurate interpolation.

2.4 Gaussian Functions

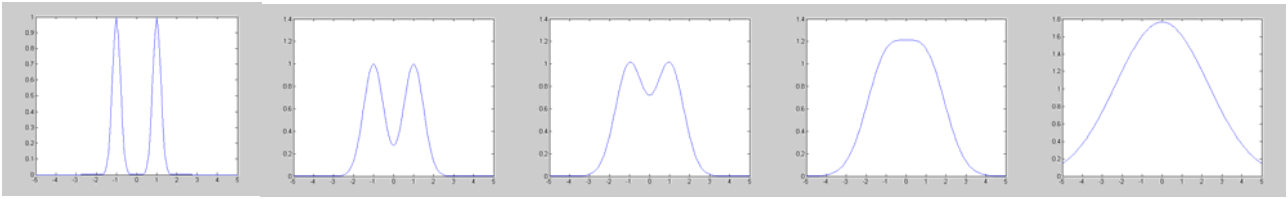
The Gaussian or normal probability density function (bell curve), used as a basis function for interpolation, tends to consistently perform well compared to other interpolation techniques [7]. A 2-D Gaussian requires six parameters for its description: There is a 2x2 symmetric covariance matrix, Σ, which determines the widths of the Gaussian, a two element mean vector μ that determines its location, and an amplitude A that determines the height of the function. Σ_{11} in the covariance matrix is the variance in the x direction, Σ_{22} is the variance in the y direction, and the two diagonal elements are the same and describe the angle of the principal axis of the 2-D Gaussian (See Equation 2.1) [2].

$$F\{x\} = \frac{A}{\sqrt{(2\pi)^2 |\Sigma|}} \exp\left[-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu) \right] \ , \ \Sigma = \begin{pmatrix} \sigma_{11}^2 \sigma_{12}^2 \\ \sigma_{21}^2 \sigma_{22}^2 \end{pmatrix} \quad (2.1)$$

2.5 Smoothness

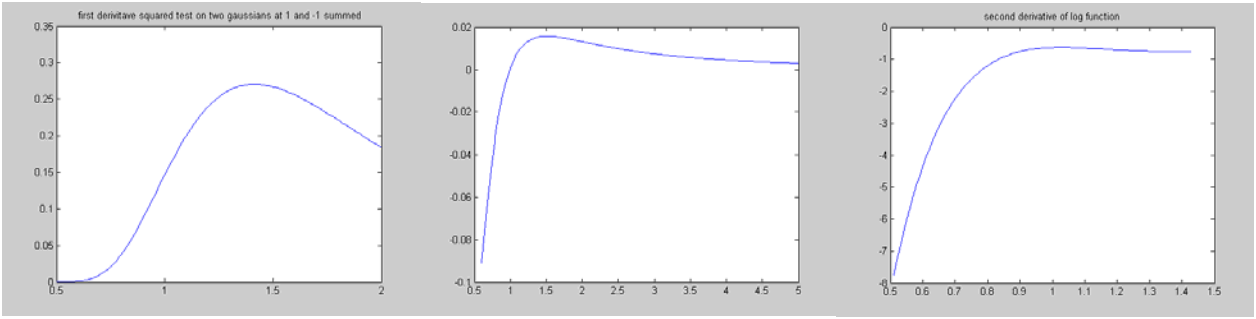
A rough image usually has sharp edges and discontinuities while a smooth image is softer with rolling edges. Removing high frequencies by low-pass filtering reduces noise in the image; however, low pass filters also blur edges [9]. Figures 6 and 7 use a toy

example to illustrate smoothness and its characterization. There are many ways to calculate smoothness, or roughness which is the inverse of smoothness, and a few are shown in the example. Figure 7a shows the result of taking the difference between pixels immediately surrounding the center of each Gaussian, and then squaring and summing the result. 7b takes a different approach by finding the second derivative of the entire curve and then squaring and summing that result. Another way to measure smoothness is to take the Integral of the log of the function multiplied by the function. This approach is shown in Figure 7c.



(a) (b) (c) (d) (e)

Figure 6 The sum of two Gaussian functions with different variances located at -1 and +1. The smoothness of the resulting function can be measured in several ways as shown in Figure 7.



(a) (b) (c)

Figure 7 Three different measures of smoothness versus the variance of the summed Gaussians in Figure 6. The sum squared difference at points 1 and -1 is shown in (a), the total integrated squared second derivative is shown in (b). and the integrated product of function and log(function) is shown in (c).

III. Methodology

Super-resolution is a process that increases the frequency content of a digital image beyond the capability of the recording device. It can be performed adaptively using the interpolation technique described here. Resolution can be increased to any degree through interpolation; however, as resolution is increased in this way, the resulting error increases as well. The goal of Adaptive Gaussian Interpolation (AGI) is to minimize this error by using the data present in the image in an effective way.

This chapter explains Adaptive Gaussian Interpolation (AGI). Figure 8 gives a brief step-by-step description of the AGI algorithm.

1. *Analyze the initial image to create a super-resolution template for the interpolated image.*
2. *Segment the image to be interpolated into 3x3 pixel windows.*
3. *Find the mean vector and covariance matrix for each window and generate the corresponding Gaussian.*
4. *Multiply all covariance matrices by a smoothness parameter.*
5. *Find the amplitude of each Gaussian such that its mean over the center pixel is the pixel gray value.*
6. *Adjust the amplitude of each Gaussian to account for the tails of neighboring Gaussians (see Figure 6).*
7. *Place the pixel gray values in a super-resolution template using the adjusted Gaussians.*

Figure 8. Recipe for the Adaptive Gaussian Interpolation (AGI) algorithm.

3.1 Acquiring/Analyzing the Image

AGI can be preformed on any image; however, due to the processor-intensive nature of the algorithm, testing is typically performed on small portions of images. Using

images on the order of 20x20 pixels, blocking artifacts to be removed using AGI image enhancement are easily recognized (See Figure 9).

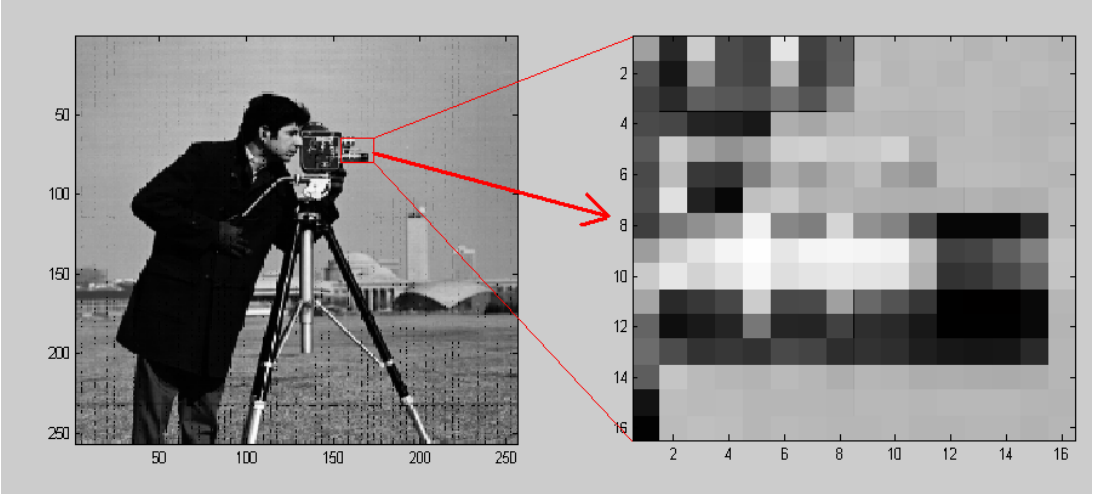


Figure 9 An example of a use of AGI, where a small portion of an image in which interesting or critical information exists is heavily magnified. The algorithm is executed on this small portion. The program takes approximately 30 seconds to interpolate the small portion of the image using a 2.8Ghz Pentium 4 processor with 1GB RAM and MATLAB software [6].

The image, once it is saved into memory, is analyzed to determine size, brightness, and smoothness. Also, all necessary input variables are read, and using this data the size of the final image is calculated. An empty template or matrix of the appropriate size is then created to hold the final image.

3.1.1 Level of Resolution Increase

Most tests are conducted using a pixel magnification of five, implying that there are five times as many pixels in the interpolated image along each axis. Thus the resulting image has 25 times the number of pixels as the original image. However, any level of resolution increase can be used.

3.1.2 Window Size

This size determines the number of pixels used to calculate the Gaussian that represents a single pixel. Using a 3x3 window, the pixel in the center of the window has a Gaussian based on nine gray levels, and a 5x5 window has a Gaussian based on 25 gray levels. This number must be odd so that there is a center pixel for the window.

3.1.3 Footprint Size

The footprint is the pixel area that each adaptive Gaussian covers in the output image. Due to the infinite nature of the Gaussian function, it must be cropped at some point. One option is to have the footprint cover the entire image; however after three standard deviations, 99% of the Gaussian is attenuated. During testing it was found that the average calculated variance is roughly 2.75. Accordingly, the footprint is limited to affect the surrounding 25 pixels. In practice it is found that more than 99% of the Gaussian is represented in most cases.

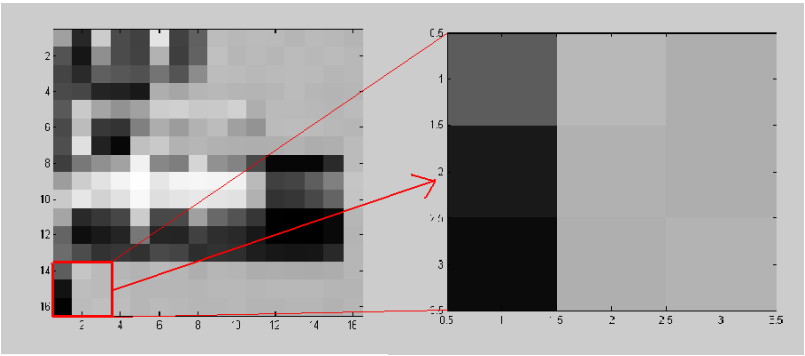


Figure 10 This figure demonstrates the method by which the image is analyzed one window at a time. A 3x3 (9 pixel) window creates a 2-D gaussian using a least squares fit. The pixel in the center of the 3x3 window is the pixel to be interpolated and replaced by a Gaussian.

3.2 Windowing the Image

In this step the image is divided into windows of a specified size. A 3x3 window is chosen for simplicity; however, any window of odd dimensions is possible (See Figure 10). The window starts at the top left pixel of the image and creates a Gaussian for the top left pixel based on the nine pixel gray values and their respective locations relative to the pixel in the center. Then the window moves down to the next pixel immediately below the first.

The pixels at the outer edge of the image do not have a complete 3x3 window of data surrounding them; to compensate, the average gray level of the entire image is calculated and used as the gray values for the missing pixels (see Figure 11).

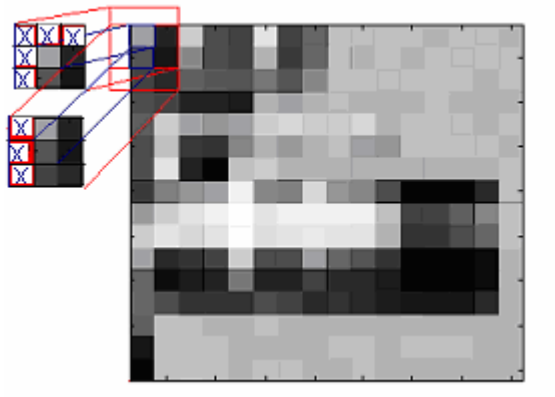


Figure 11 The Xed pixels represent locations where data is needed but not present. The mean image gray level is substituted for the missing data.

3.3 Calculating the Basis Function

The adaptive basis function is calculated using a least squares fit to the data in the 3x3 window; equations are shown below. First, the weighted mean of the window is calculated based on the gray level values and their distances from the center of the window using Equation 3.1. The weighted mean vector (U_x,U_y) is calculated by weighting distance from the window center with the pixel gray values ($P_{(1-9)}$). Distances are calculated using the coordinate system as shown in Figures 13 and 14. The gray level data in each window is normalized to one before being extracted using the format in Figure 12. U_x is the location of weighted mean in the x direction and U_y is the location of the weighted mean in the y direction. (U_x,U_y) then make up the mean vector for the 2-D Gaussian. Each pixel value is treated as a point located in the center of the pixel. Data for a 9x2 matrix is calculated by multiplying the gray value of the pixels with the x distance from the mean vector. This data is entered into the first column of the 9x2 matrix and the same is done using the y distance for the second column. This matrix is then pre-multiplied by its transpose to generate a 2x2 covariance matrix (see Equation 3.3). Figure 14 shows the output Gaussian for the data in Figures 12 and 13.

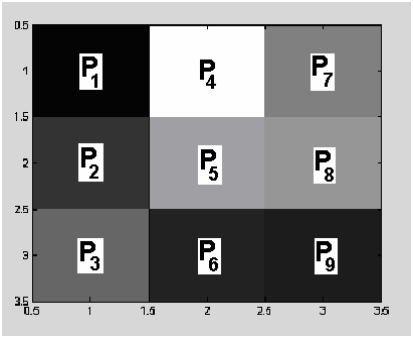


Figure 12 This figure shows how the pixels are counted linearly in MATLAB and for the algorithm. In Equations 3.1 and 3.2 the terms P1-P9 represent the pixels labeled 1-9 respectively in the image above.

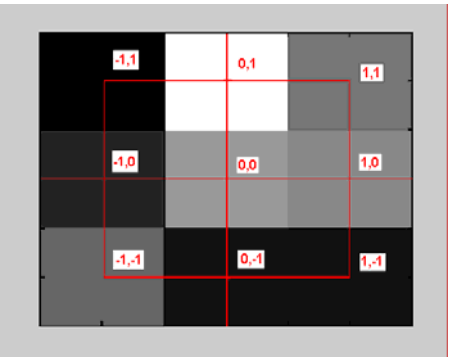


Figure 13 Window coordinate system, in which dark colors represent low numbers and lighter colors represent higher numbers; thus the data is centered slightly to the top right of the center of the window.

$$U_x = ((-1)\cdot P_1)+(0\cdot P_4)+(1\cdot P_7) + ((-1)\cdot P_2)+(0\cdot P_5)+(1\cdot P_8) + ((-1)\cdot P_3)+(0\cdot P_6)+(1\cdot P_9) \quad (3.1a)$$

$$U_y = (1\cdot P_1)+(1\cdot P_4)+(1\cdot P_7) + (0\cdot P_2)+(0\cdot P_5)+(0\cdot P_8) + ((-1)\cdot P_3)+((-1)\cdot P_6)+((-1)\cdot P_9) \quad (3.1b)$$

$$CovMatrix = \begin{bmatrix} P_1(-1-U_x) & P_2(-1-U_x) & P_3(-1-U_x) & P_4(0-U_x) & P_5(0-U_x) & P_6(0-U_x) & P_7(1-U_x) & P_8(1-U_x) & P_9(1-U_x) \\ P_1(1-U_y) & P_2(0-U_y) & P_3(-1-U_y) & P_4(1-U_y) & P_5(0-U_y) & P_6(-1-U_y) & P_7(1-U_y) & P_8(0-U_y) & P_9(-1-U_y) \end{bmatrix} \bullet \begin{bmatrix} P_1(-1-U_x) & P_1(1-U_y) \\ P_2(-1-U_x) & P_2(0-U_y) \\ P_3(-1-U_x) & P_3(-1-U_y) \\ P_4(0-U_x) & P_4(1-U_y) \\ P_5(0-U_x) & P_5(0-U_y) \\ P_6(0-U_x) & P_6(-1-U_y) \\ P_7(1-U_x) & P_7(1-U_y) \\ P_8(1-U_x) & P_8(0-U_y) \\ P_9(1-U_x) & P_9(-1-U_y) \end{bmatrix} \quad (3.2)$$

For the 5x5 window, the same coordinate system is used as is for the 3x3 case only it is extended (see figure 14 and Equation 3.3). As a result, those gray levels further from the center of the window have the most influence.

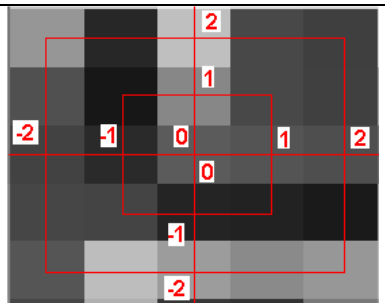


Figure 14 Coordinate system for the 5x5 window size

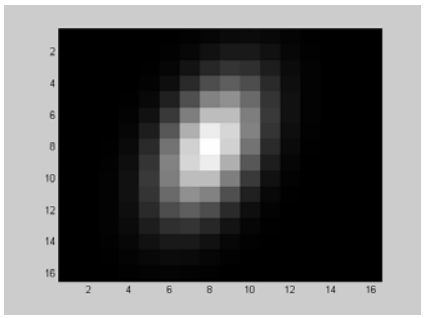


Figure 15 Gaussian created from the data in Figure 13 using the covariance matrix calculated in Equation (3.3).

$$
\begin{aligned}
U_x = {} & ((-2)\cdot P_1)+((-1)\cdot P_6)+(0\cdot P_{11})+(1\cdot P_{16})+(2\cdot P_{21}) \\
& +((-2)\cdot P_2)+((-1)\cdot P_7)+(0\cdot P_{12})+(1\cdot P_{17})+(2\cdot P_{22}) \\
& +((-2)\cdot P_3)+((-1)\cdot P_8)+(0\cdot P_{13})+(1\cdot P_{18})+(2\cdot P_{23}) \\
& +((-2)\cdot P_4)+((-1)\cdot P_9)+(0\cdot P_{14})+(1\cdot P_{19})+(2\cdot P_{24}) \\
& +((-2)\cdot P_5)+((-1)\cdot P_{10})+(0\cdot P_{15})+(1\cdot P_{20})+(2\cdot P_{25})
\end{aligned}
\qquad (3.3a)
$$

$$
\begin{aligned}
U_y = {} & (2\cdot P_1)+(2\cdot P_6)+(2\cdot P_{11})+(2\cdot P_{16})+(2\cdot P_{21}) \\
& +(1\cdot P_2)+(1\cdot P_7)+(1\cdot P_{12})+(1\cdot P_{17})+(1\cdot P_{22}) \\
& +(0\cdot P_3)+(0\cdot P_8)+(0\cdot P_{13})+(0\cdot P_{18})+(0\cdot P_{23}) \\
& +((-1)\cdot P_4)+((-1)\cdot P_9)+((-1)\cdot P_{14})+((-1)\cdot P_{19})+((-1)\cdot P_{24}) \\
& +((-2)\cdot P_5)+((-2)\cdot P_{10})+((-2)\cdot P_{15})+((-2)\cdot P_{20})+((-2)\cdot P_{25})
\end{aligned}
\qquad (3.3b)
$$

3.3.1 Smoothness

After the covariance matrix is created, a covariance multiplier is needed to determine smoothness. AGI uses a predetermined multiplier designed to work for all images. This number is calculated by sampling images of different resolutions and recording their respective smoothness (see Figure 16).

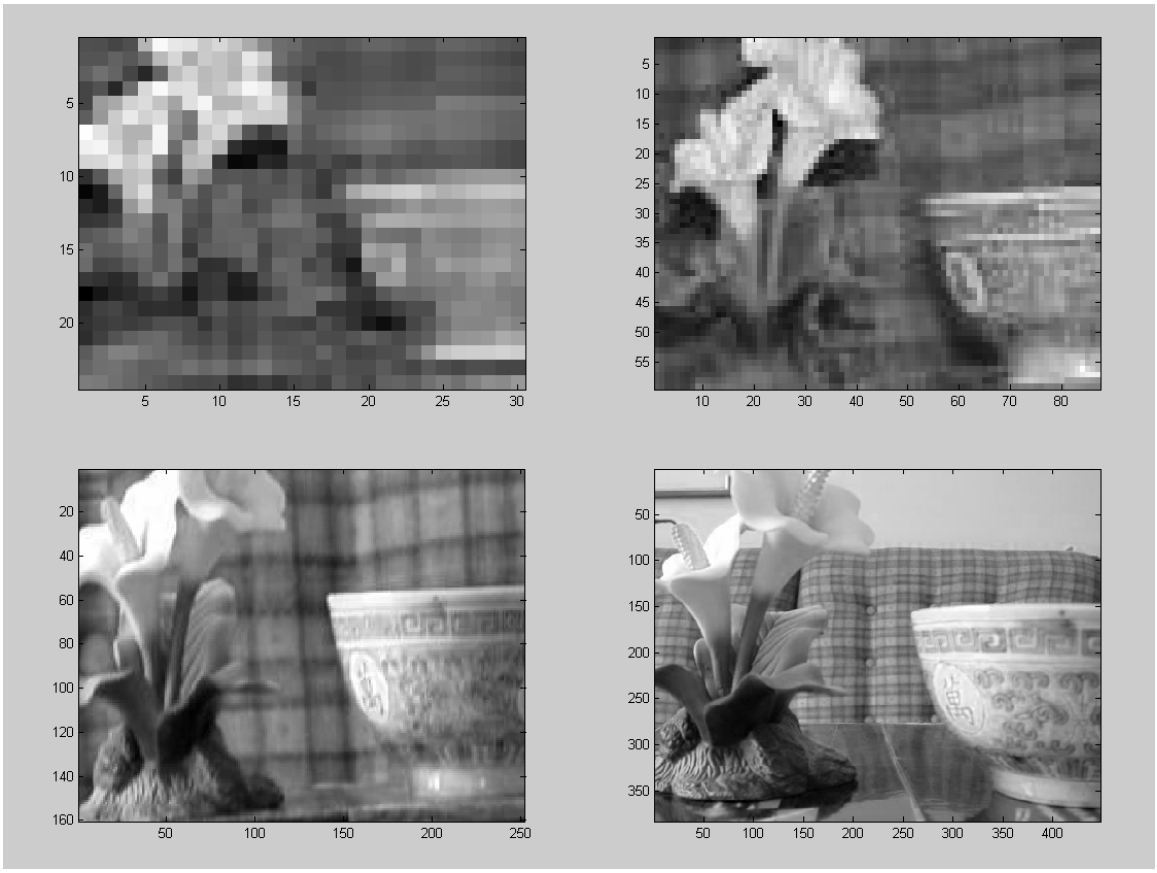


Figure 16 Images recorded at different resolutions using a digital camera for data source. This data was used to create the chart in Figure 17.

Smoothness in this case is determined using a mean second derivative squared of the entire image matrix. It is found (generally) that as resolution increases, the smoothness per pixel increases as well. This relationship is used to calculate the

smoothness of the resulting super-resolved image. The corresponding covariance matrix multiplier is applied to match the estimated smoothness (see Figure 17).

3.3.2 Initial Magnitude Calculation

The final parameter needed to create a custom 2-D Gaussian for the center window pixel is the Gaussian magnitude. To determine this value the constraint was used that the integral of the interpolated pixels over the area of the original pixel equals the gray level of the original pixel. To meet this requirement, a default value of 1 is used for the magnitude and the integral is calculated and divided by 25 (the total number of pixels). Then mean gray level over the original pixel area is set equal to pixel gray level.

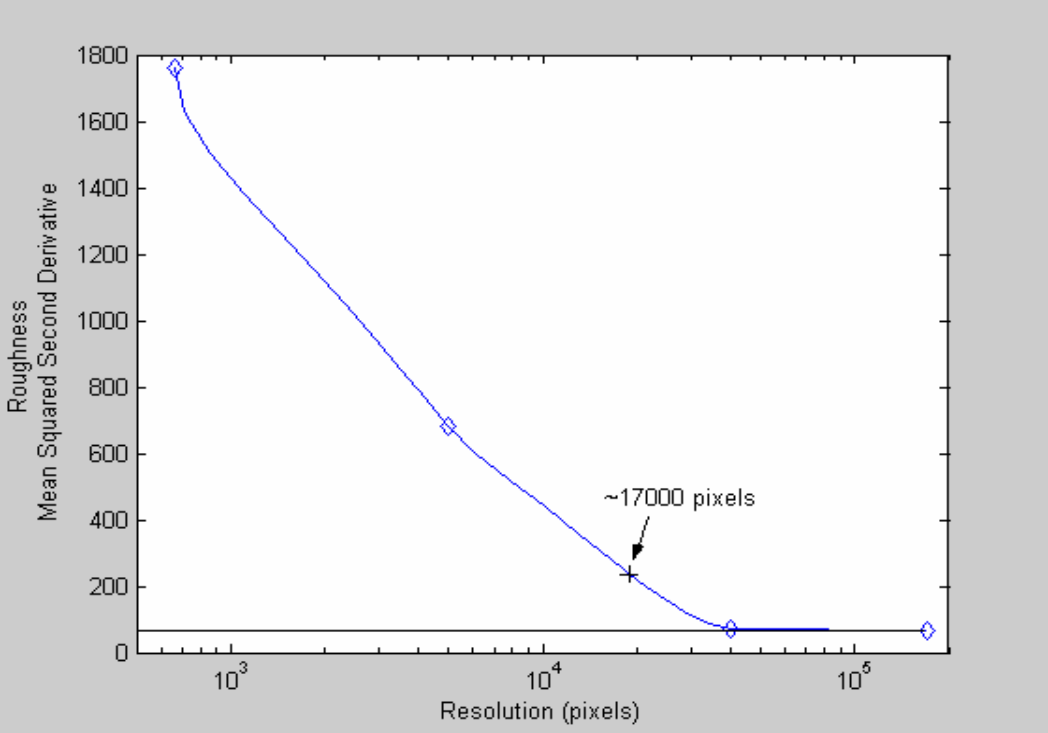
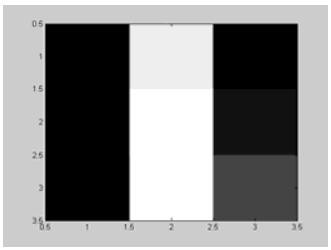


Figure 17. Calculation of roughness for use in the Adaptive Gaussian Interpolation algorithm. The curve was created using the four distance-degraded images of Figure 5. The y axis measures roughness and is the sum squared second derivative for all pixels divided by the number of pixels. As resolution increases roughness converges to about 65. A cubic equation fit to the four points is shown. The roughness from this curve corresponding to an image of about 17000 pixels is indicated, which is used in the AGI algorithm.

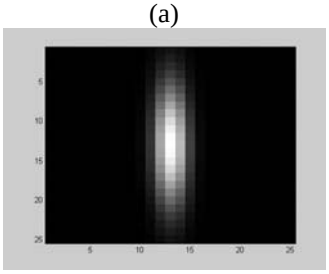
At this point the custom 2-D Gaussian for the window is completely described and the program moves to the next pixel until every pixel has an associated Gaussian. Examples of two different Gaussians are shown in Figures 18 and 19.

3.4 Resampling

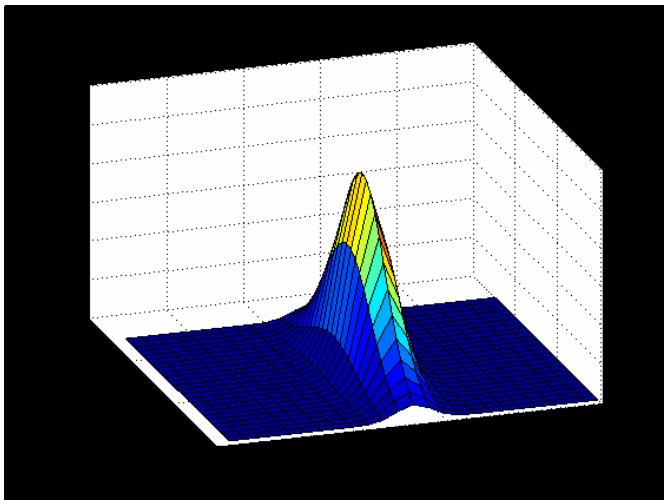
As each Gaussian is calculated, it is sampled at the desired rate and added to the blank template. Each Gaussian is added to its respective location centered on the original pixel that it replaces. Using this adding technique, all Gaussians significantly overlap. A simple demonstration of this smoothing effect using three curves is shown in Figure 20.



(a)

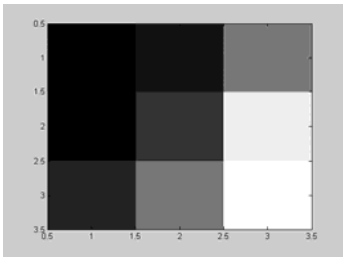


(b)

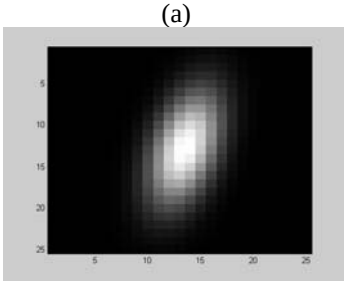


(c)

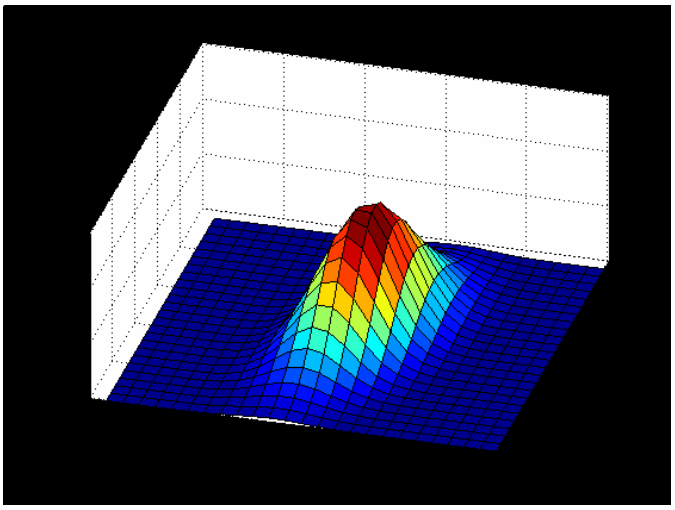
Figure 18. Gaussian interpolation using surrounding pixels. (a) The pixel in the center is interpolated. The values of all 9 pixels, interpreted as points at the pixel centers, are fit to a Gaussian using least squares. (b) Image of the resulting Gaussian for the center pixel. (c) A 3-D plot of the resulting Gaussian.



(a)

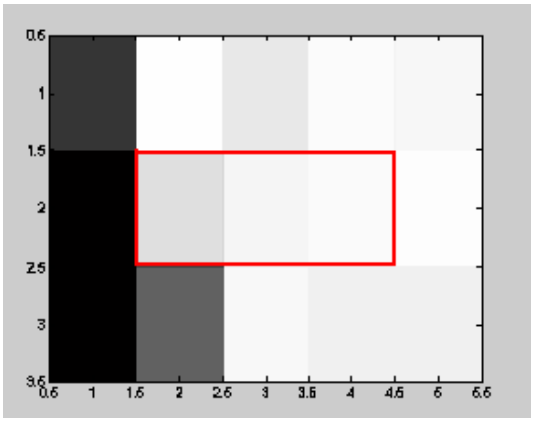


(b)

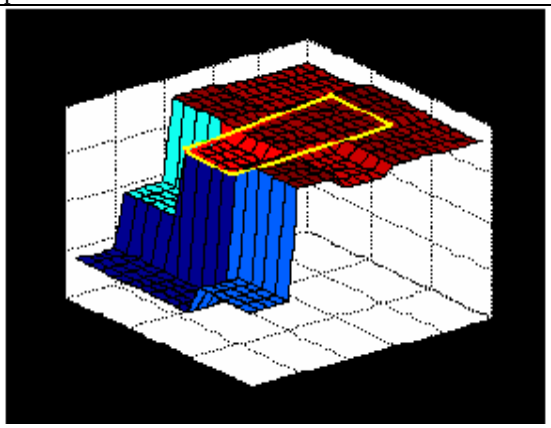


(c)

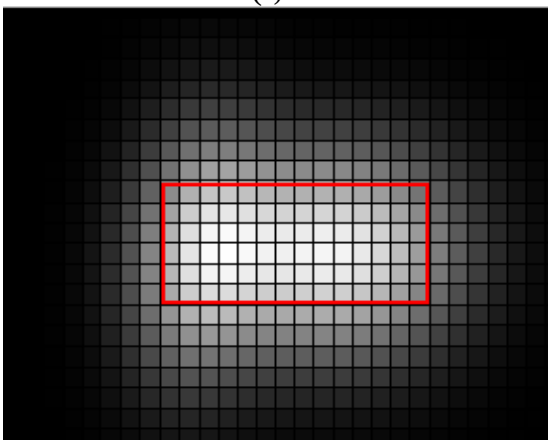
Figure 19. As in Figure 17, but for a smoother group of pixels.



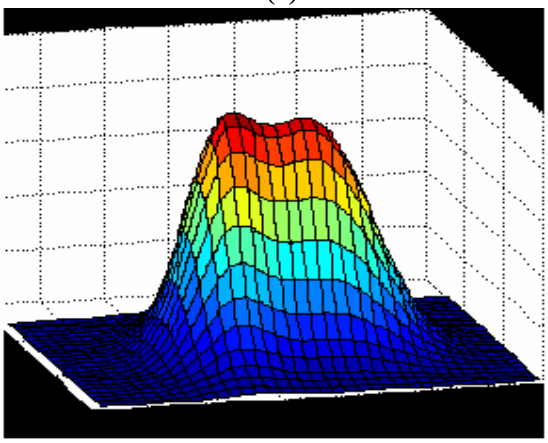
(a)



(b)



(c)



(d)

Figure 20 The addition of three interpolated pixels is shown in this figure: (a) shows the three interpolated pixels inside a box. Each pixel is interpolated separately and then the three resulting Gaussians are added to obtain the result in (c); also, (b) and (d) show 3-D visualizations of (a) and (c), respectively.

3.5 Final Magnitude Calculation

After all Gaussians are added to the blank template, the program recalculates the magnitude of each individual Gaussian to compensate for the influence of the tails from adjacent Gaussians. In order to accurately meet the constraint placed on the program, i.e., the average gray level of the image over the original pixel is the same as the original pixel gray level, an iterative method is implemented. This method is time consuming and does not significantly affect the image after the first iteration; however, depending on desired accuracy when meeting the constraint, several iterations may be required. A diagram for the method is shown in Figure 21.

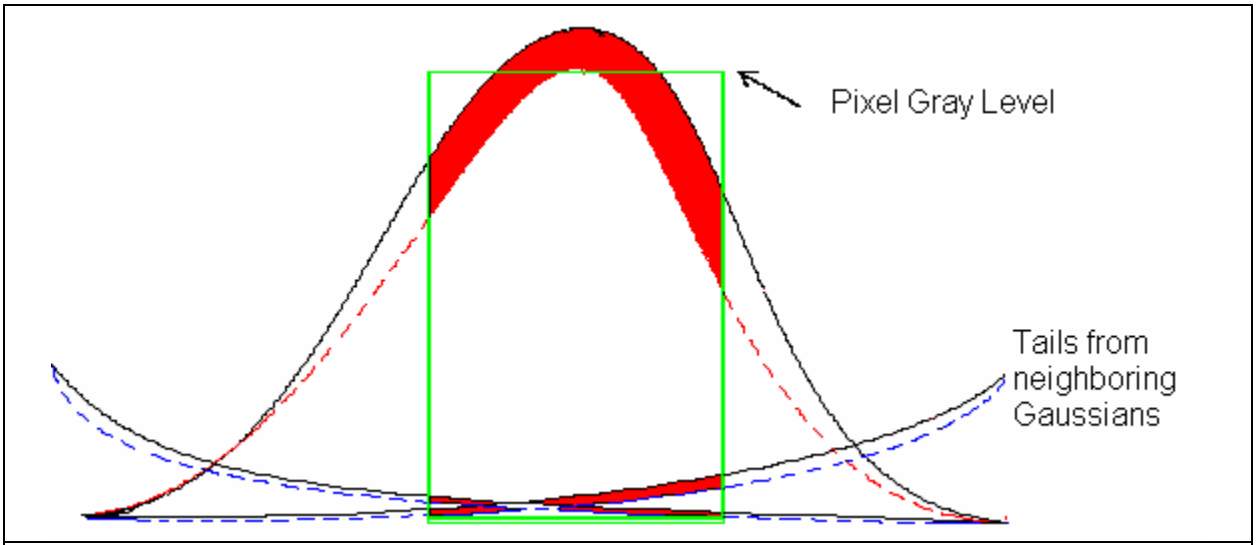


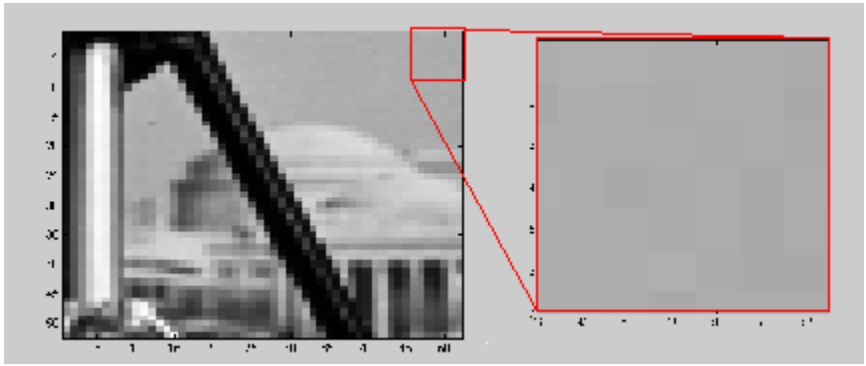
Figure 21. The Adaptive Gaussian Interpolation algorithm requires that the mean of all micro-pixels covering the area of each original pixel have the gray level of that pixel. This requirement is realized by an iterative process in which the tails of neighboring pixels are taken into account. The solid lines are the original Gaussian and tails from neighboring Gaussians, and the dashed lines are the adjusted Gaussian and neighboring tails. The shaded area is lost during the adjustments.

3.6 Limiting Brightness

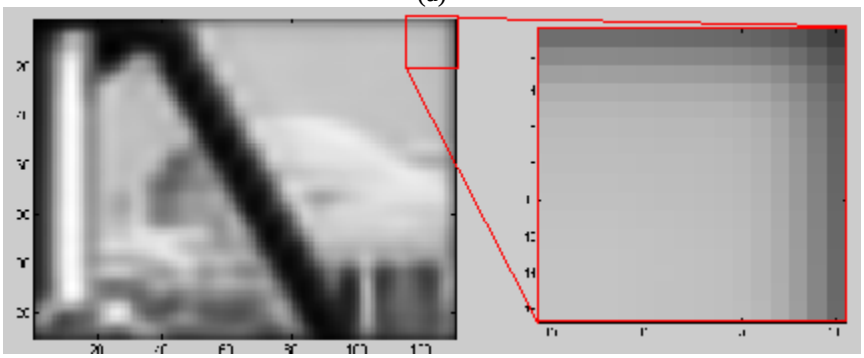
When all data has been calculated and added into the final output matrix, the brightness of the image is adjusted by limiting the highest value pixel. There are often extreme cases which result in unrealistically high values for a few pixels, and because the program maximizes contrast by setting the lowest pixel value to the darkest black and the highest to the brightest white, an outlying high value can darken the rest of the image. To correct this problem a maximum pixel value is set to adjust the outlying high values. This is done by matching the mean gray level value of the original image to that of the interpolated image which ensures that the pixel values of the images are on the same scale. Then the maximum pixel value is set to equal the maximum pixel value from the original image. This ensures that the brightness of the interpolated image is similar to that of the original. Due to the increased smoothness of the interpolated image it was found that setting the interpolated maximum brightness to approximately 1.2 times the original maximum pixel brightness most closely represents an accurate interpolation.

3.7 Final Cropping

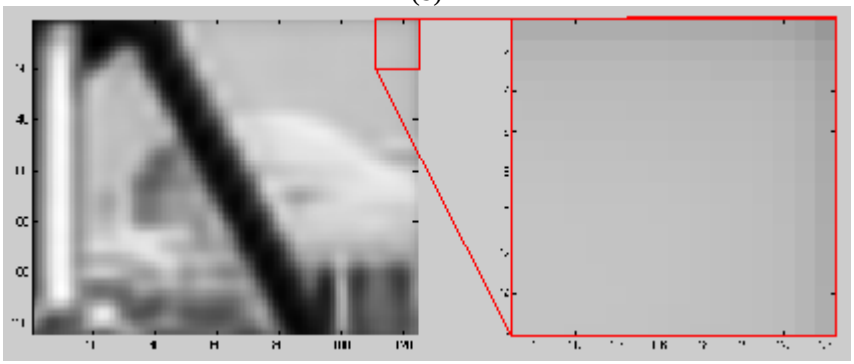
The filled super-resolution template is cropped to cut off the tails of the Gaussians on the edges, which allows the resulting image to have the required 25 times the number of original pixels. Because the outer Gaussian are not smoothed by outlying tails, the image can be further cropped to eliminate a slight fading artifact. Figure 22 illustrates this artifact.



(a)



(b)



(c)

Figure 22. Edge fading artifact: (a) is the original image, (b) shows fading at the edge of the image after AGI has been executed, (c) demonstrates how cropping the outer four pixels removes this effect; however image data is also lost.

3.8 Determining the Best Variables

After program completion, the best input variables are determined in order to accurately compare results. A resolution enhancement on the order of five (total resolution increase of 25) is used as the default to determine the optimum window size. The smoothness criterion is also checked for desirable results. The two metrics used to determine output quality are the mean square error in pixel gray level and a subjective analysis.

3.8.1 Window Size

Two window sizes are tested: 5x5 and 3x3. It is found that the 3x3 window size gives superior results according to both metrics. It is assumed that the performance decrease continues as window size increases. Results are shown in Figures 23 and 24.

3.8.2 Smoothness Verification

The two metrics give conflicting results in this case: mean square error analysis indicates that increased smoothness has reduced error, but subjective analysis agrees with the roughness analysis described earlier in this chapter. Thus the latter method is selected. The results are shown in Figures 25 and 26.

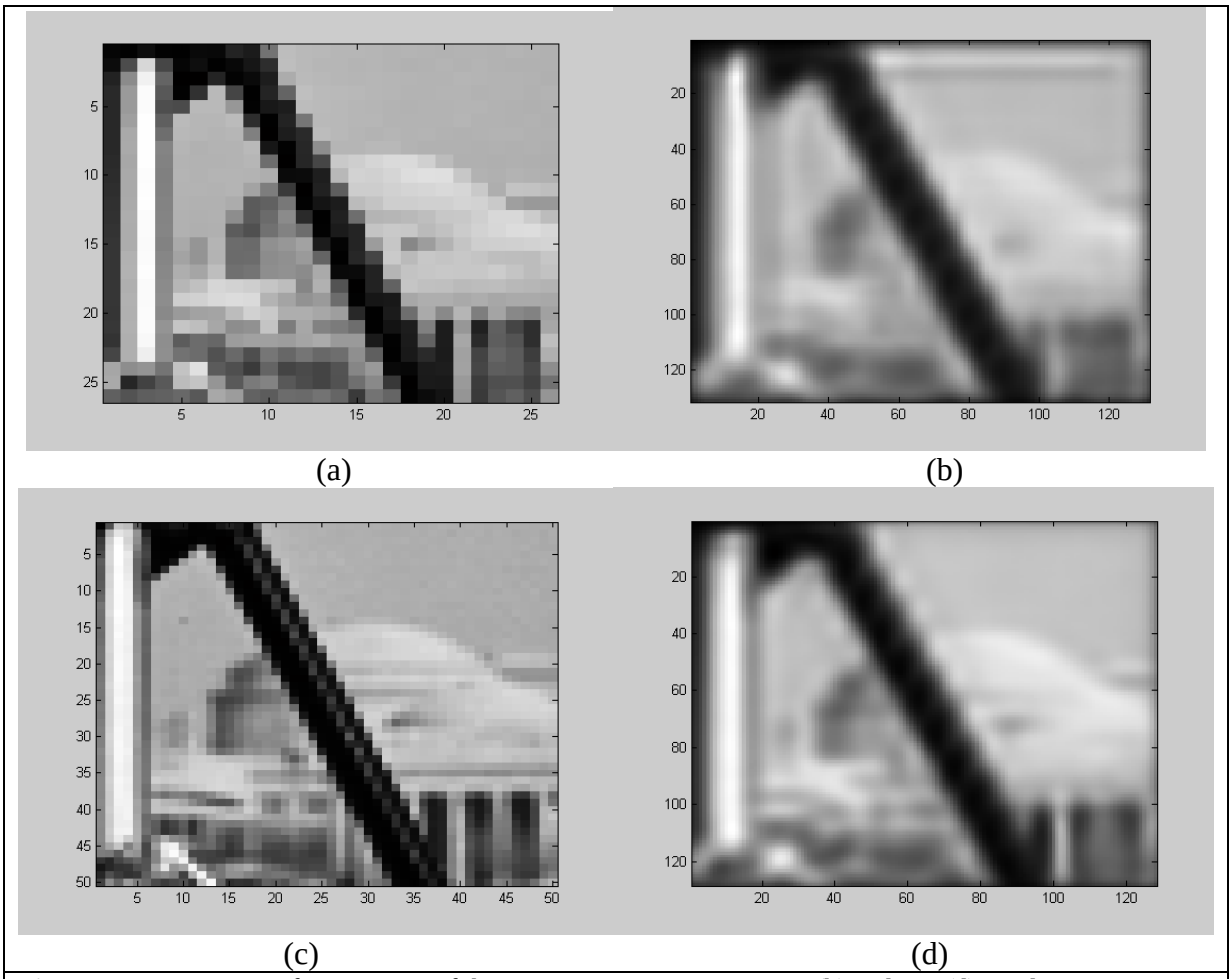


Figure 23 Comparison of AGI output of the cameraman image using 5x5 (b) and 3x3 (d) window sizes. Here (a) is the input image and (b) is the original maximum resolution image. It is concluded that the 3x3 window gives superior results compared to the 5x5 window.

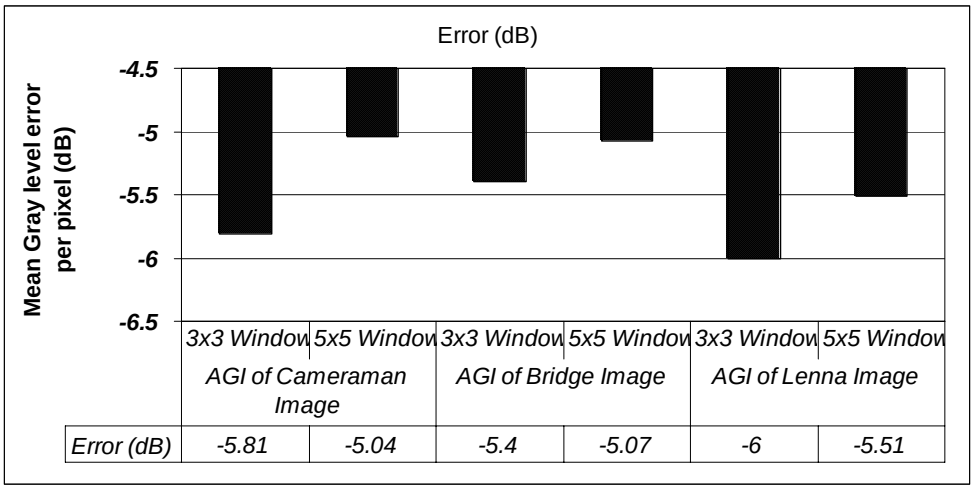
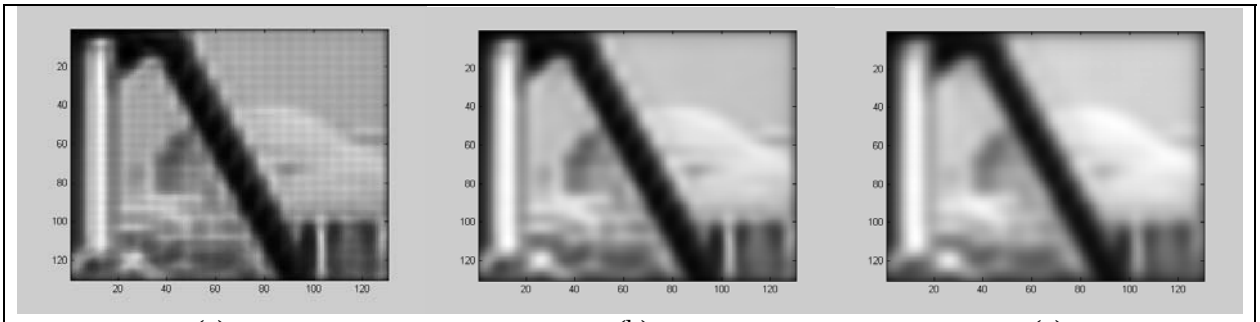


Figure 24 Mean square error in pixel gray level for 3x3 and 5x5 window sizes. It is found that the 3x3 window consistently outperforms the 5x5 window using this metric. Error is measured with respect to maximum error. Maximum error is defined as no data or a pure black image (zero matrix).



(a) (b) (c)

Figure 25 Demonstration of the effect of the covariance multiplier on image output. The images above have covariance multiplier values of (a) 80, (b) 140, and (c) 240. As expected, the larger variance images exhibit more blurring effects, whereas low variance retains blocking phenomena.

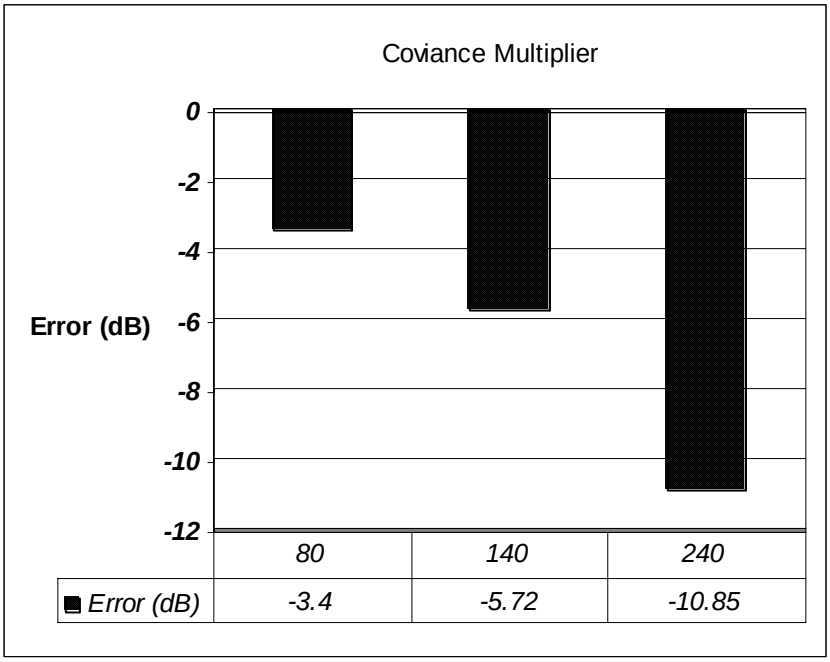


Figure 26 Mean square error in AGI images with different covariance multipliers. This comparison shows that error is reduced as smoothness increases.

IV Results

This chapter focuses on analyzing the results of AGI execution and comparing them with other interpolation techniques using both objective and subjective metrics. Images are also transformed into the frequency domain for further comparison. Finally, the images are subjected to various edge detection algorithms similar to those used in computer vision.

The two main comparison techniques are subjective comparison between the images and an objective method using the mean square gray level error of the interpolated image versus the original image. The output images from the AGI algorithm use the input variables shown in Figure 27. These variables were determined to be desirable as shown in Chapter 3.

Interpolation Technique	Level of Resolution enhancement (per dimension)	Variance	Footprint size (In terms of original pixels)	Window Size	Covariance Multiplier
AGI	5	variable	5x5	3x3	140
GRBF	5	2.7	5x5	1	n/a
Bicubic	5	n/a	n/a	n/a	n/a
Bilinear	5	n/a	n/a	n/a	n/a

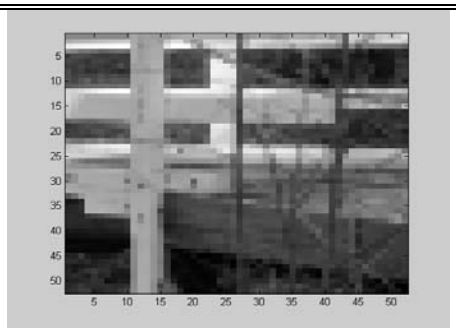
Figure 27 This figure shows the variables used to generate the results. Level of resolution enhancement determines the relative number of pixels used to describe the interpolated image relative to the original (5 per dimension implies 25 total). The smoothness setting is derived from the smoothness curve in Figure 16.

4.1 The Test Images

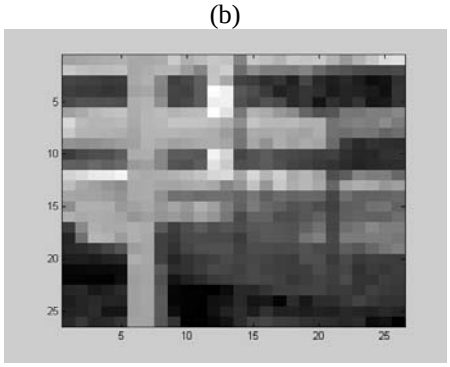
The test images are chosen to represent several commonly found artifacts. Bridge, Cameraman, and Lenna are commonly used images that represent a range of image characteristics. The airplane and stadium images are satellite photos, which could become

a common application for the AGI technique. All images (see Figures 28-32) are down sampled after low pass filtering in order to simulate natural image degradation.

(a)

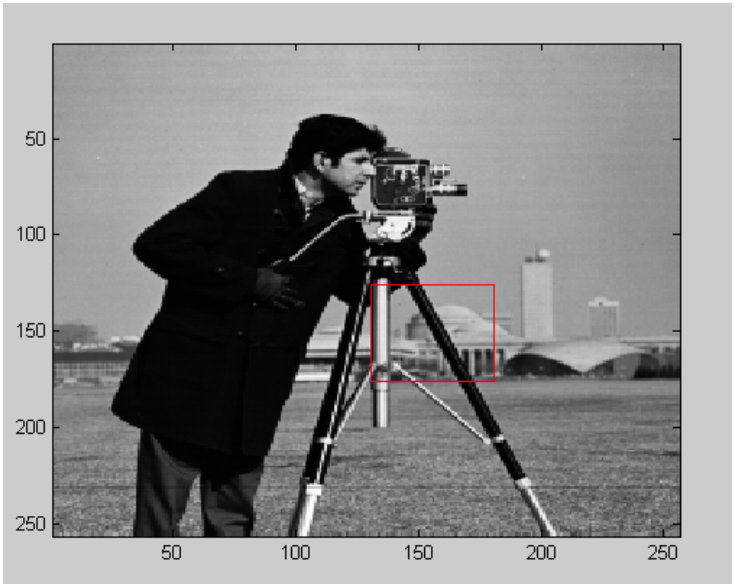


(b)

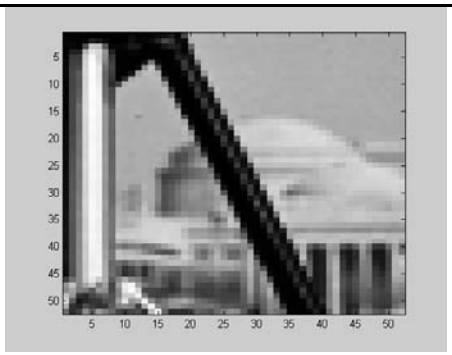


(c)

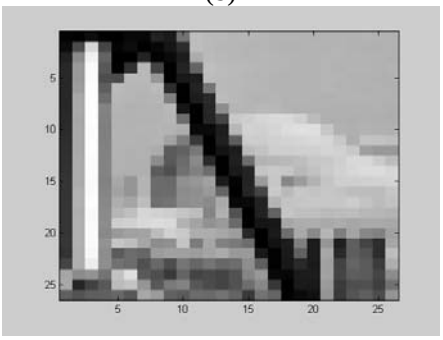
Figure 28 Bridge image (a) with the interpolated portion boxed,(b) with this portion enlarged, (c) with this portion low-pass filtered and down sampled.



(a)



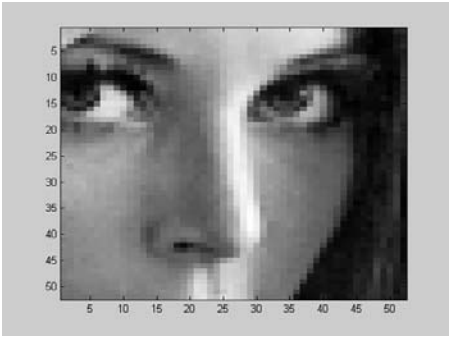
(b)



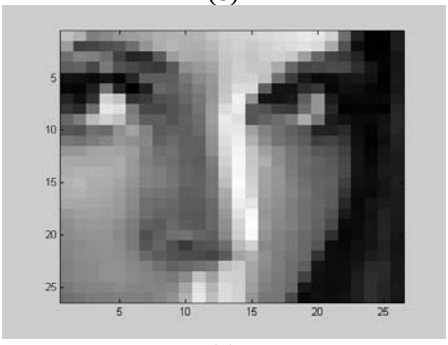
(c)

Figure 29 Cameraman image (a) with the interpolated portion boxed, (b) with this portion enlarged, (c) with this portion low pass filtered and down sampled.

(a)



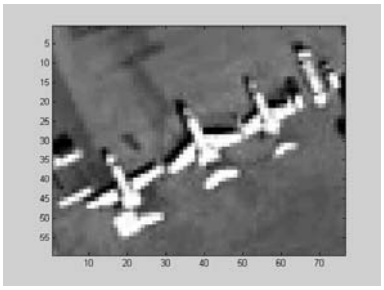
(b)



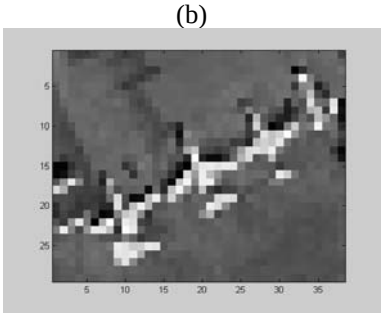
(c)

Figure 30 Lenna image (a) with the interpolated portion boxed, (b) with this portion enlarged, (c) with this portion low pass filtered and down sampled.

(a)

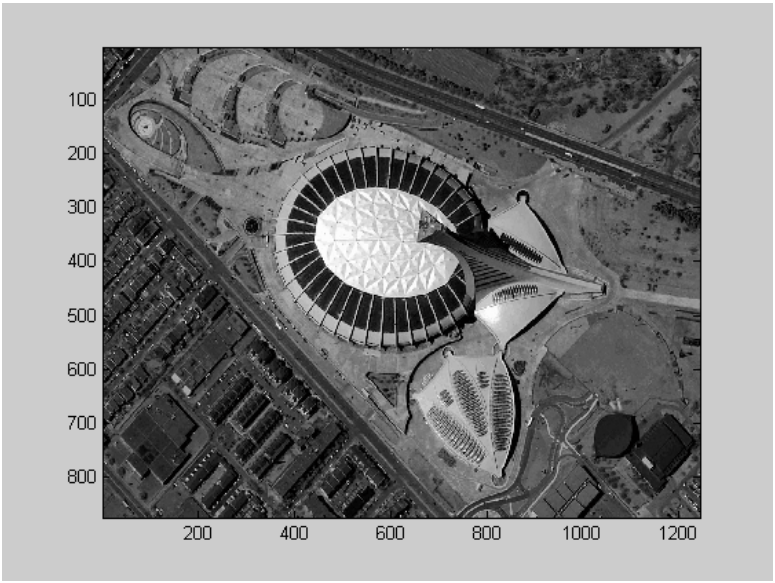


(b)

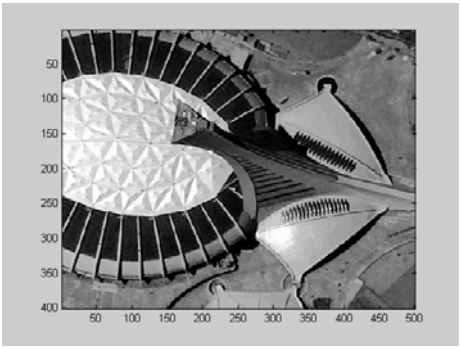


(c)

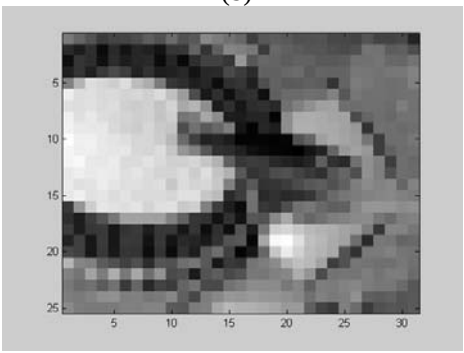
Figure 31 Satellite image of airplanes (a) with the interpolated portion boxed, (b) with this portion enlarged, (c) with this portion low pass filtered and down sampled.



(a)



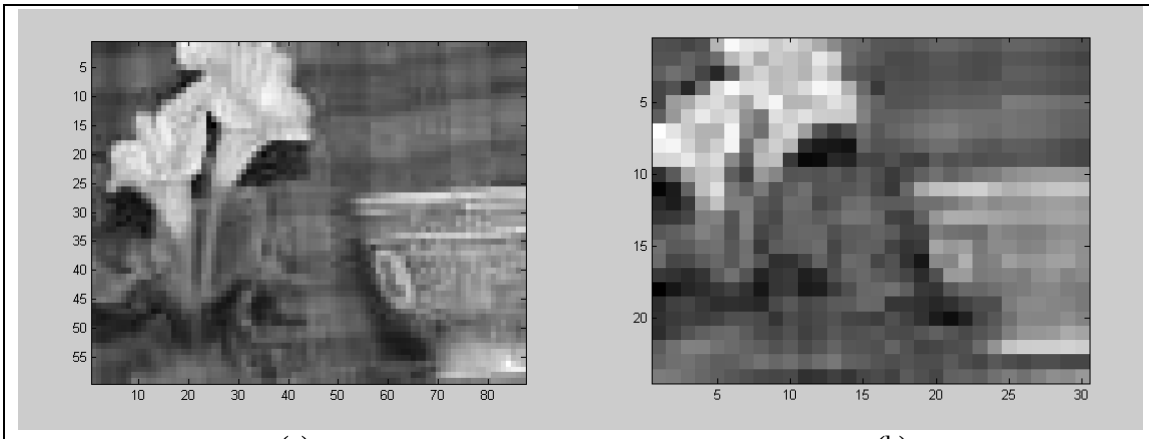
(b)



(c)

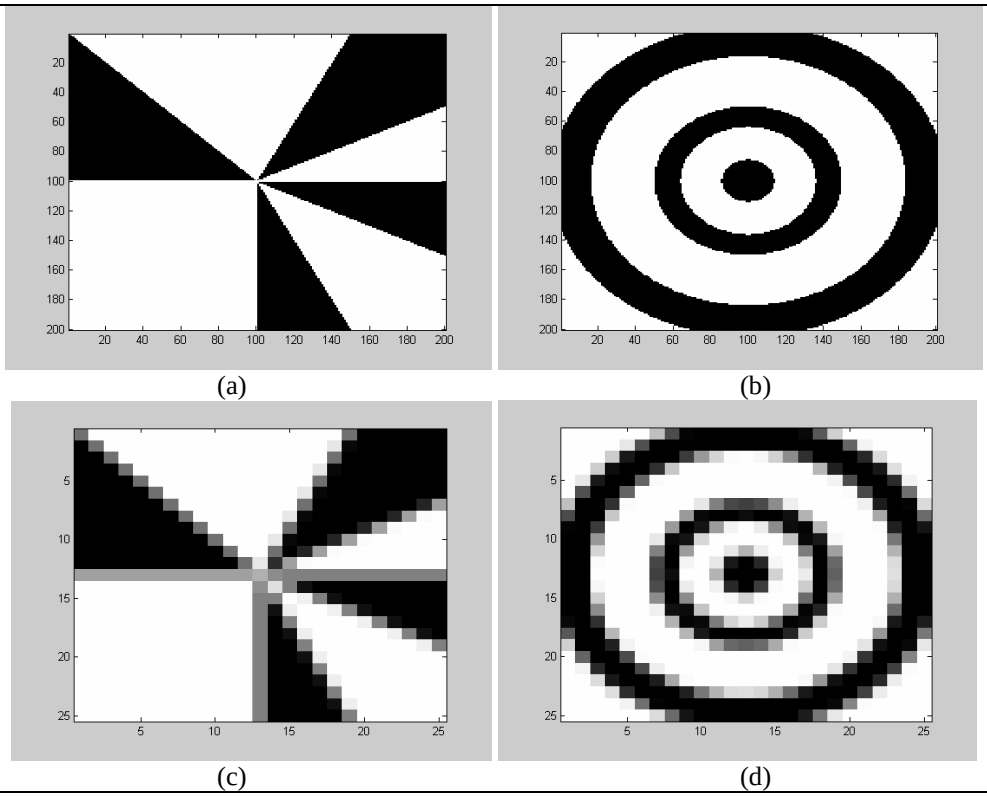
Figure 32 (a) Satellite image of a stadium, (b) Enlarged portion to be interpolated, (c) with this portion low pass filtered and down sampled.

The flower and bowl image in Figure 33 was taken with a digital camera from varying distances to simulate real image degradation. The geometric shape images are artificially created to help demonstrate strengths the and weaknesses of each interpolation method. These images and their degraded versions are shown in Figures 33 and 34.



(a) (b)

Figure 33 Images recorded at different resolutions using a digital camera. (a) shows a higher resolution image of (b), which is the image to be interpolated.



(a) (b)

(c) (d)

Figure 34 Geometric shape test images (a) and (b), where (c) and (d) are down-sampled versions.

4.2 Subjective Test Results

Figures 35-42 show results from execution of AGI as well as equivalent results from competing interpolation techniques. Each figure shows the following:

1. The original high resolution image
2. The reduced resolution image to be interpolated
3. Output of the AGI algorithm
4. Output of Gaussian radial basis function interpolation
5. Output of bicubic Interpolation
6. Output of bilinear Interpolation

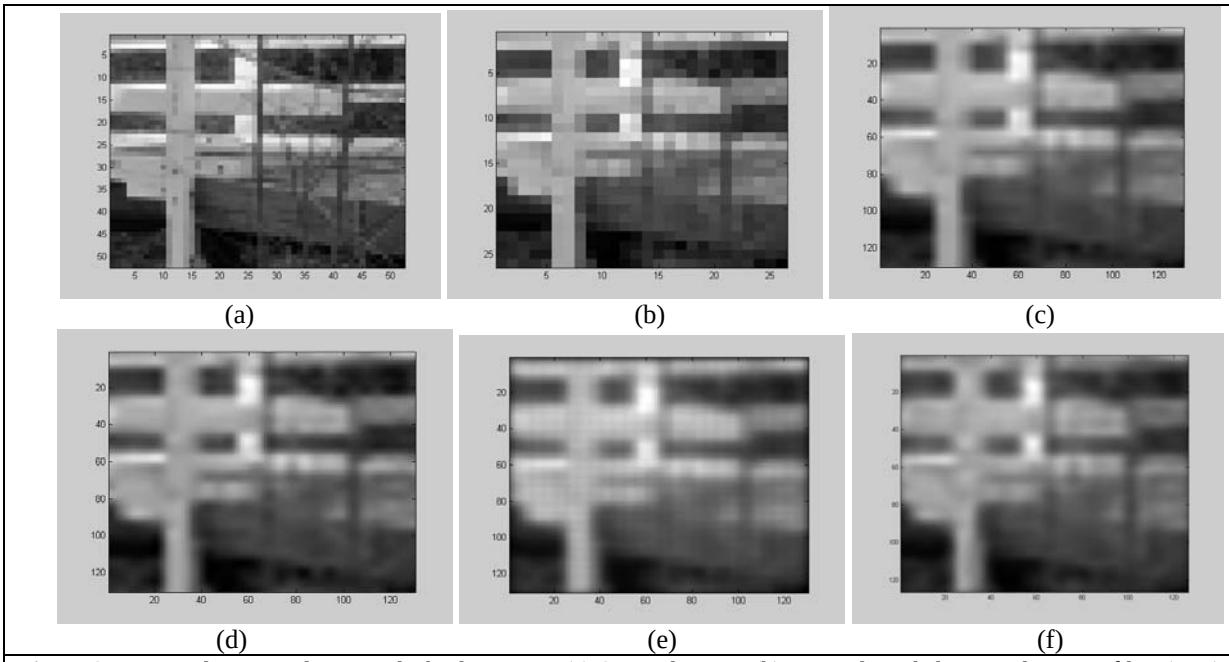


Figure 35. Interpolation results using the bridge image. (a) Original image, (b) Image degraded using a low pass filter (LPF), (c) Bilinear interpolation, (d) Bicubic interpolation, (e) Gaussian radial basis function interpolation, (f) Adaptive Gaussian Interpolation (AGI).

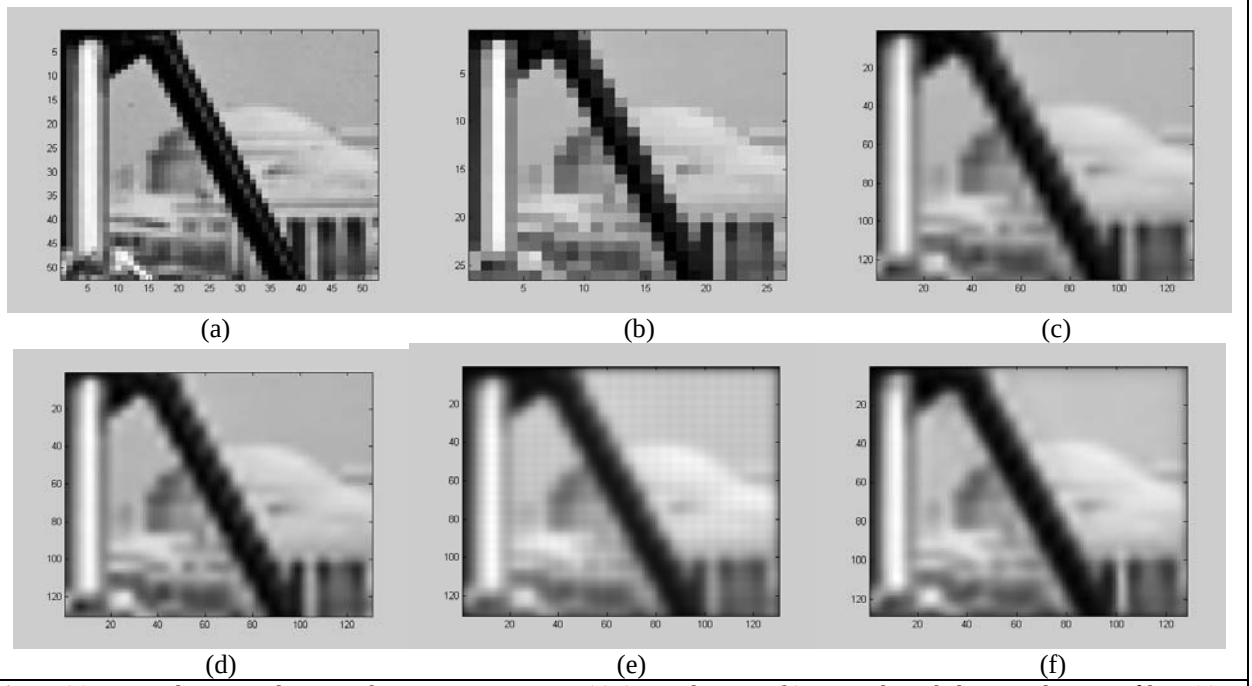


Figure 36. Interpolation results using the cameraman image. (a) Original image, (b) Image degraded using a low pass filter, (c) Bilinear interpolation, (d) Bicubic interpolation, (e) Gaussian radial basis function interpolation, (f) Adaptive Gaussian Interpolation.

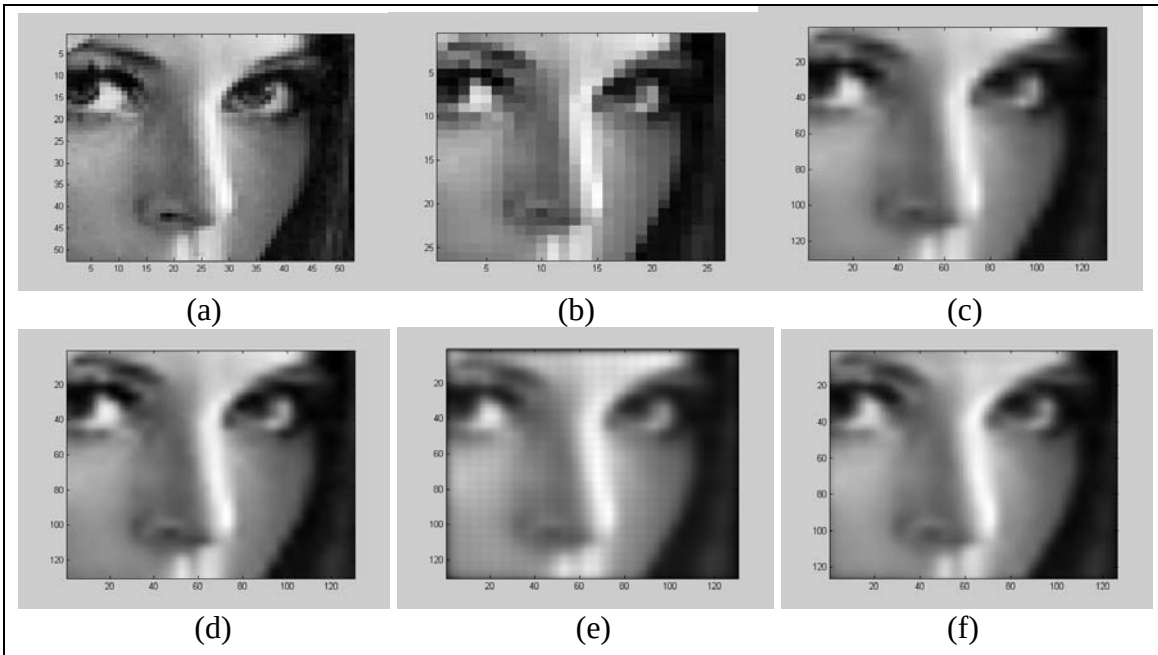
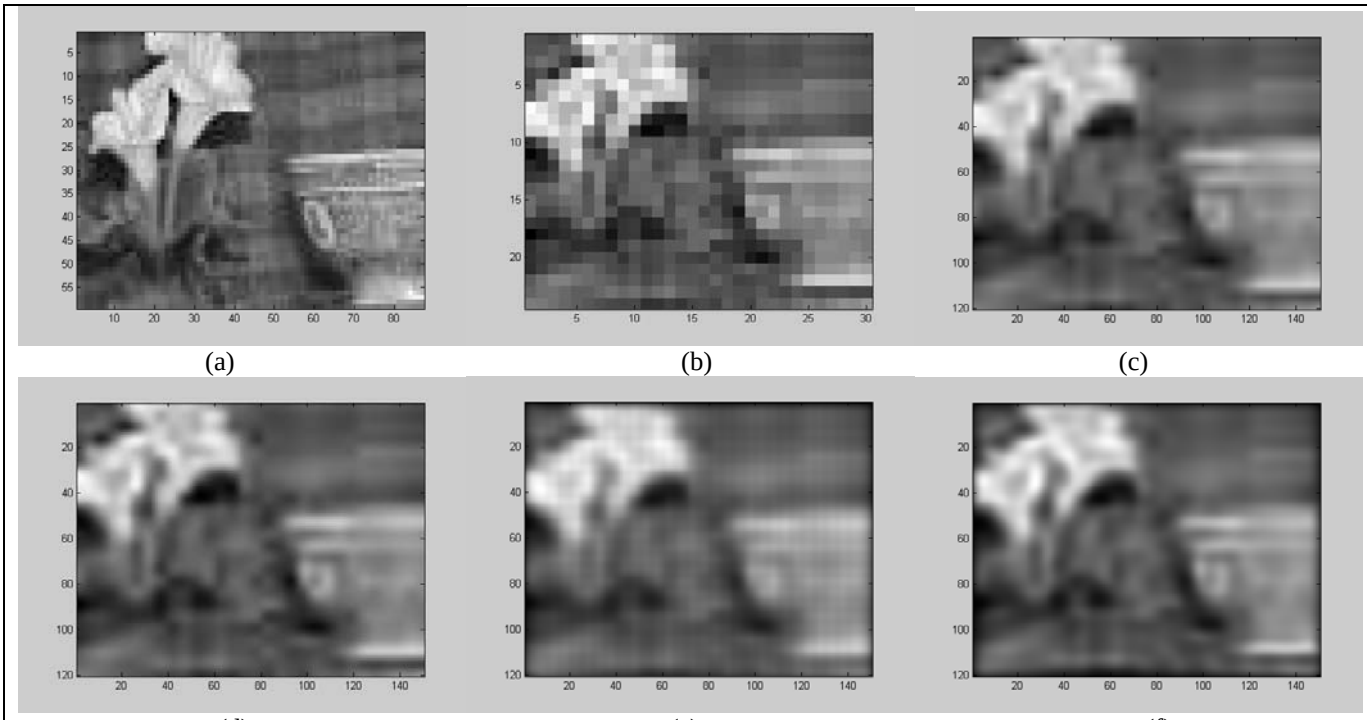


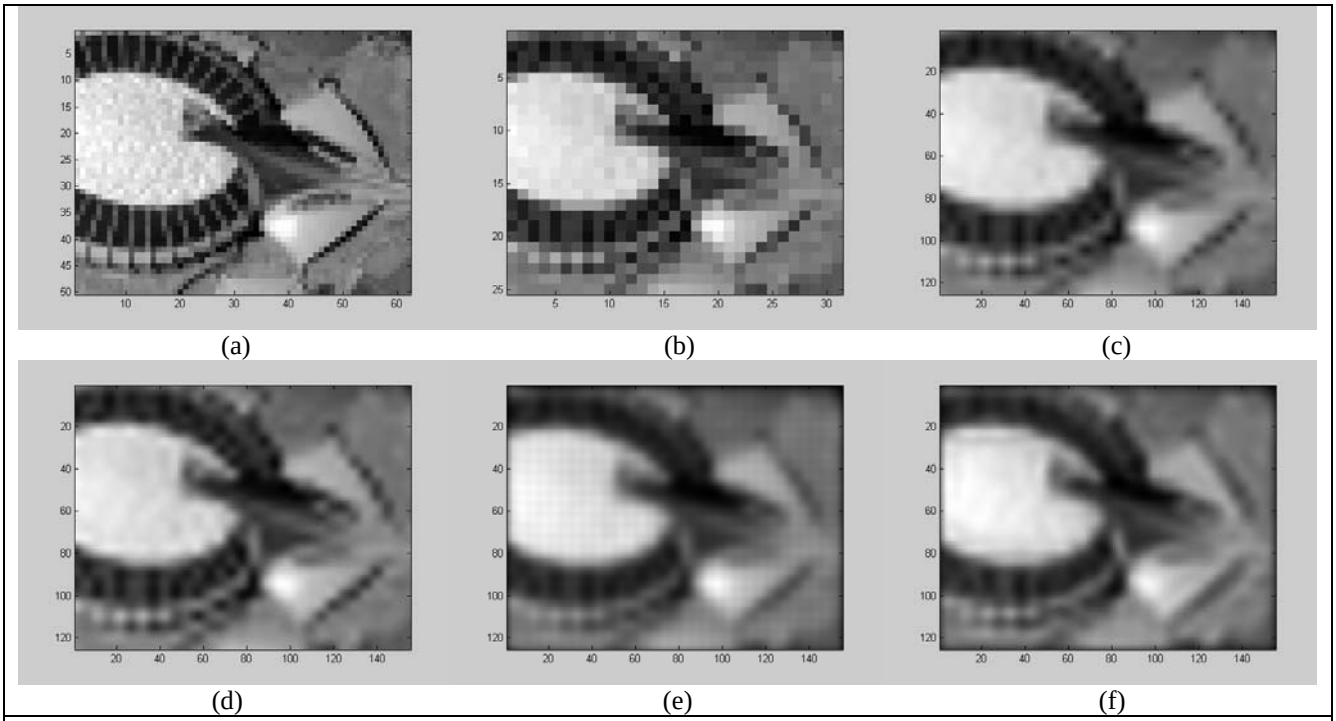
Figure 37. Interpolation results using the Lenna image. (a) Original image, (b) Image degraded using a low pass filter, (c) Bilinear interpolation, (d) Bicubic interpolation, (e) Gaussian radial basis function interpolation, (f) Adaptive Gaussian Interpolation.



(a) (b) (c)

(d) (e) (f)

Figure 38. Interpolation results using the Flower and Bowl image. (a) original high resolution image, (b) Image recorded from a greater distance, (c) Bilinear interpolation, (d) Bicubic interpolation, (e) Gaussian radial basis function interpolation, (f) Adaptive Gaussian Interpolation.



(a) (b) (c)

(d) (e) (f)

Figure 39. Interpolation results using the Stadium image. (a) Original image, (b) Image degraded using a low pass filter, (c) Bilinear interpolation, (d) Bicubic interpolation, (e) Gaussian radial basis function interpolation, (f) Adaptive Gaussian Interpolation.

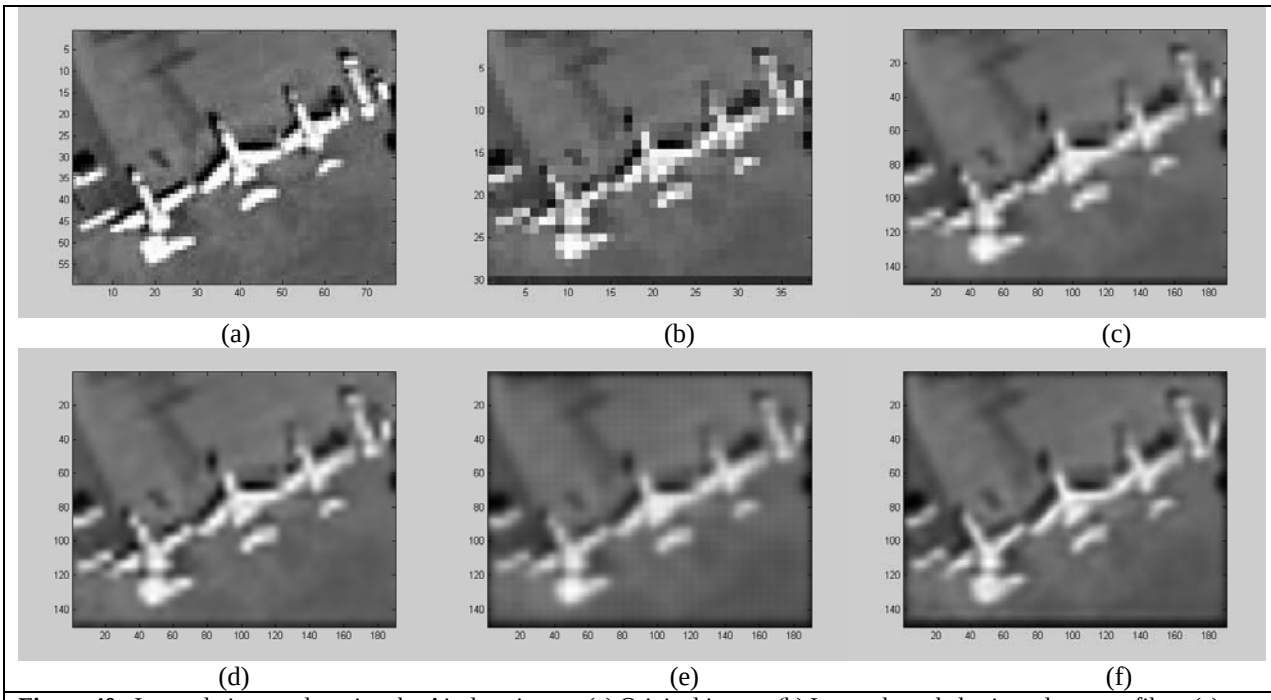


Figure 40. Interpolation results using the Airplane image. (a) Original image, (b) Image degraded using a low pass filter, (c) Bilinear interpolation, (d) Bicubic interpolation, (e) Gaussian radial basis function interpolation, (f) Adaptive Gaussian Interpolation.

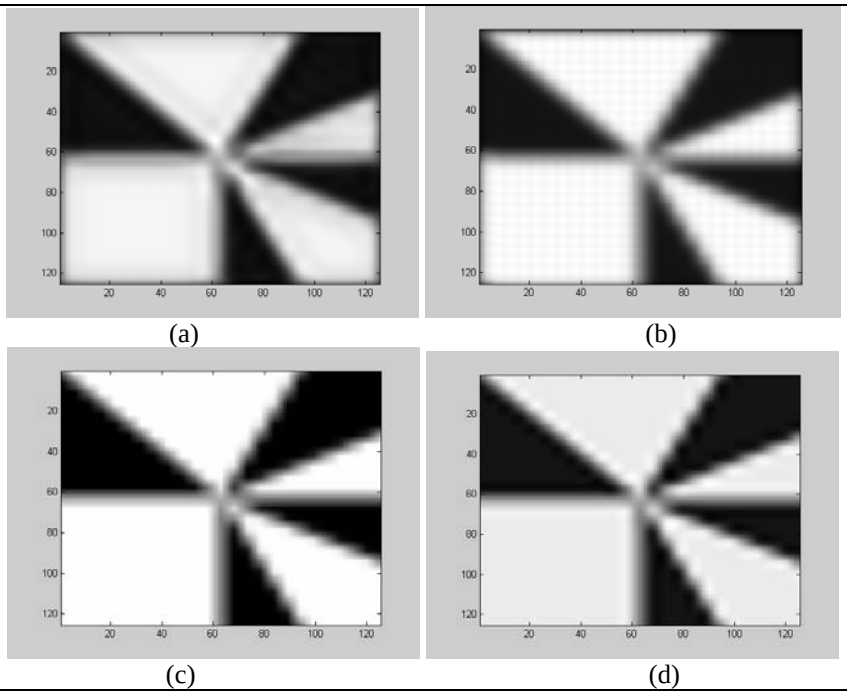


Figure 41 Interpolation of the geometric design image in Figure 33. (a) is the image after Adaptive Gaussian Interpolation, (b) is the image after Gaussian radial basis function interpolation, (c) is the image after bilinear interpolation, and (d) is the image after bicubic interpolation.

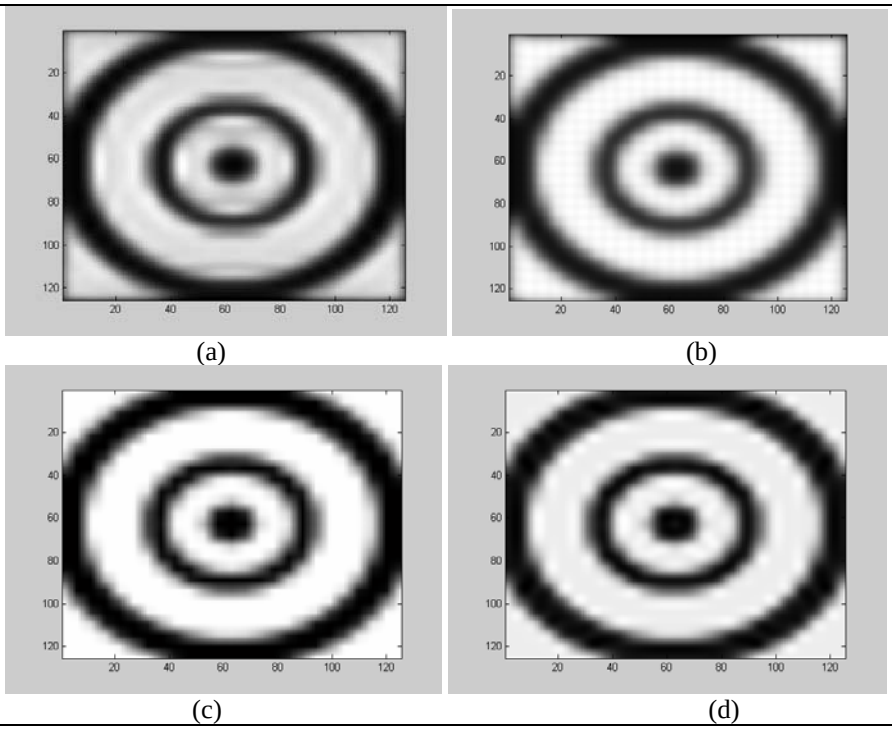
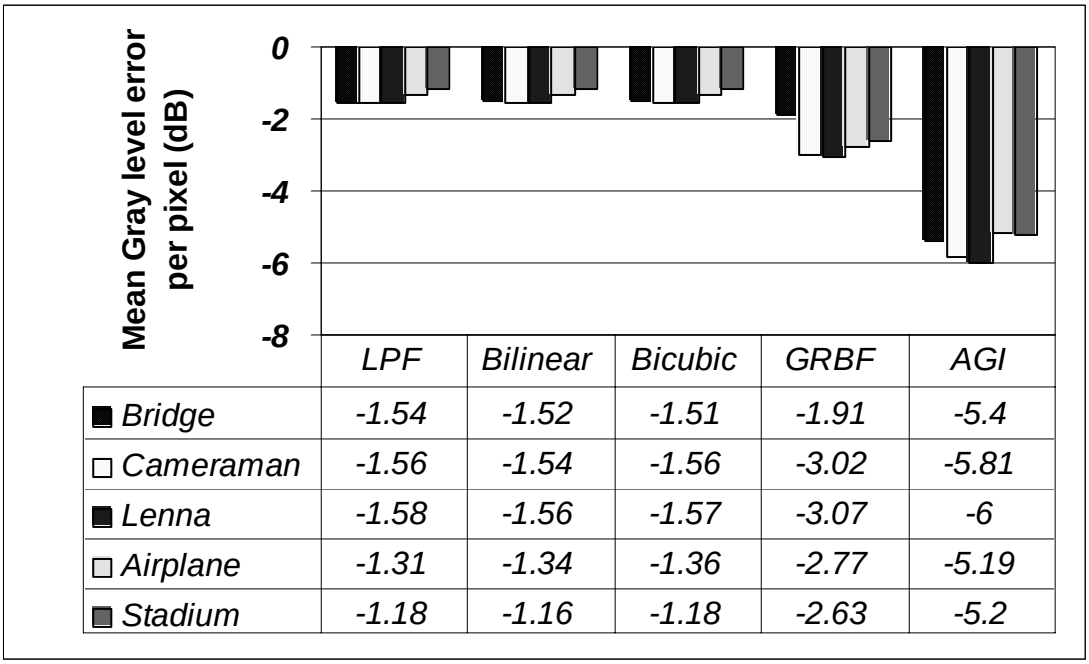


Figure 42 Interpolation of the concentric circles image in Figure 33. (a) is the image after Adaptive Gaussian Interpolation. (b) is the image after Gaussian radial basis function interpolation. (c) is the image after bilinear interpolation, and (d) is the image after bicubic interpolation.

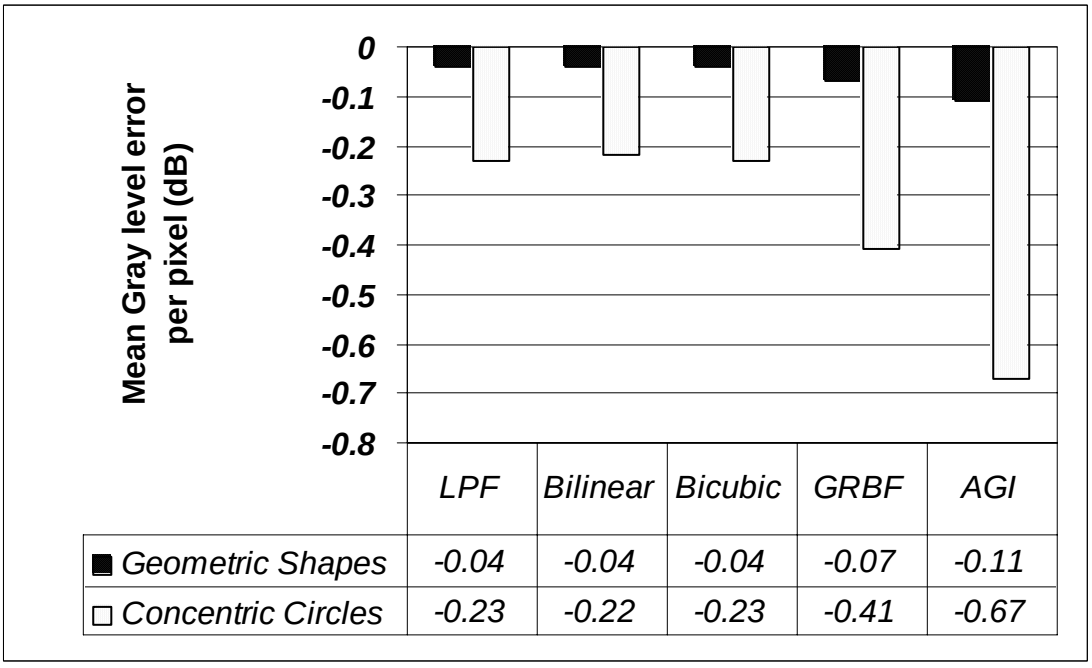
4.3 Objective Test Results

The output from each interpolation technique is analyzed using an objective metric. The mean square error in the pixel gray level is measured and the results are shown in the following figures. This error is calculated by taking the mean difference in the pixels values between the interpolated image and the original image. This analysis is not possible for the flower and bowl image due to the method by which is was captured.



	LPF	Bilinear	Bicubic	GRBF	AGI
Bridge	-1.54	-1.52	-1.51	-1.91	-5.4
Cameraman	-1.56	-1.54	-1.56	-3.02	-5.81
Lenna	-1.58	-1.56	-1.57	-3.07	-6
Airplane	-1.31	-1.34	-1.36	-2.77	-5.19
Stadium	-1.18	-1.16	-1.18	-2.63	-5.2

Figure 43 Mean square gray level error for test images. Error is measured against the original image. All interpolation is performed on a Low Pass Filtered (LPF) version of the original image, which has one-fourth the resolution of the original image.

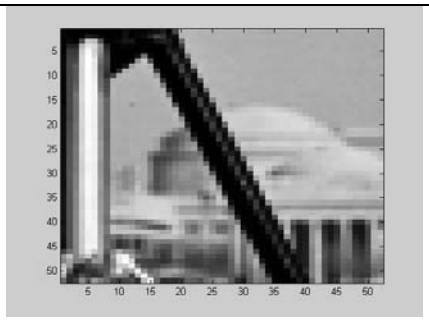


	LPF	Bilinear	Bicubic	GRBF	AGI
Geometric Shapes	-0.04	-0.04	-0.04	-0.07	-0.11
Concentric Circles	-0.23	-0.22	-0.23	-0.41	-0.67

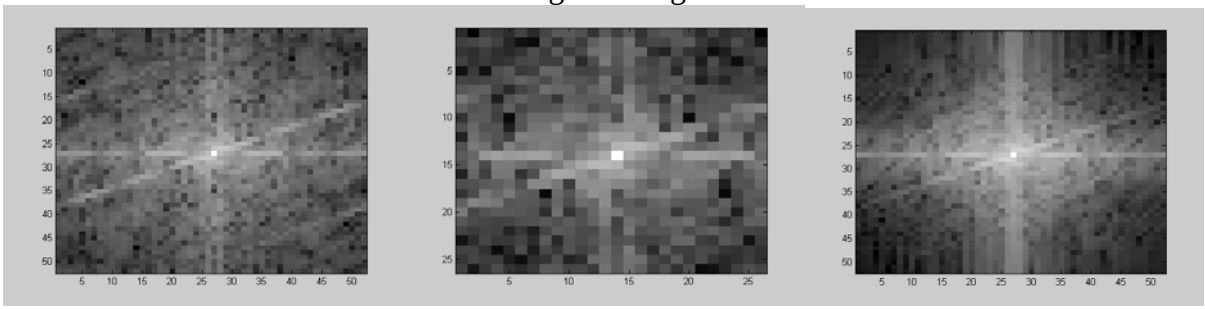
Figure 44 As in Figure 43 but for geometric images.

4.4 Frequency Extrapolation

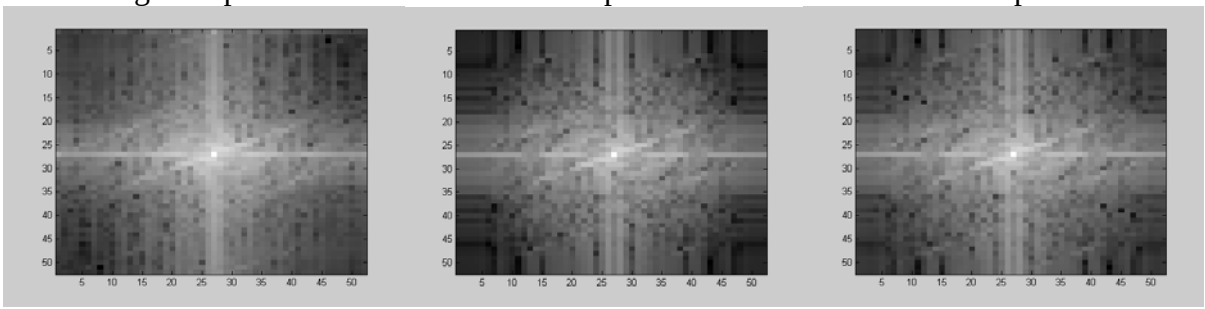
The frequency extrapolation capability of the AGI algorithm is analyzed using the Fourier transform of the output images by comparing results with the Fourier transform of the original image. In order to allow a subjective comparison, all displayed frequency spectra are the natural log of the Fourier transform. In Figure 45, all displays are cropped to show the same frequency spectrum as the original image, excluding the low pass filtered image, which has less frequency content. The complete frequency spectrum of all interpolation outputs is shown in Figure 46. Note that the larger amount of high frequency information in the bilinear and GRFB spectra primarily results from the blocking artifacts that are maintained in these interpolation methods. The mean square error in the pixel gray levels is shown in Figure 47.



Original Image



Original Spectrum | LPF Spectrum | AGI Spectrum



GRBF Spectrum | Bicubic Spectrum | Bilinear Spectrum

Figure 45 Comparison of cropped spectra of interpolated images. The full spectra of the interpolated images are shown in Figure 50. All images are logarithmically displayed.

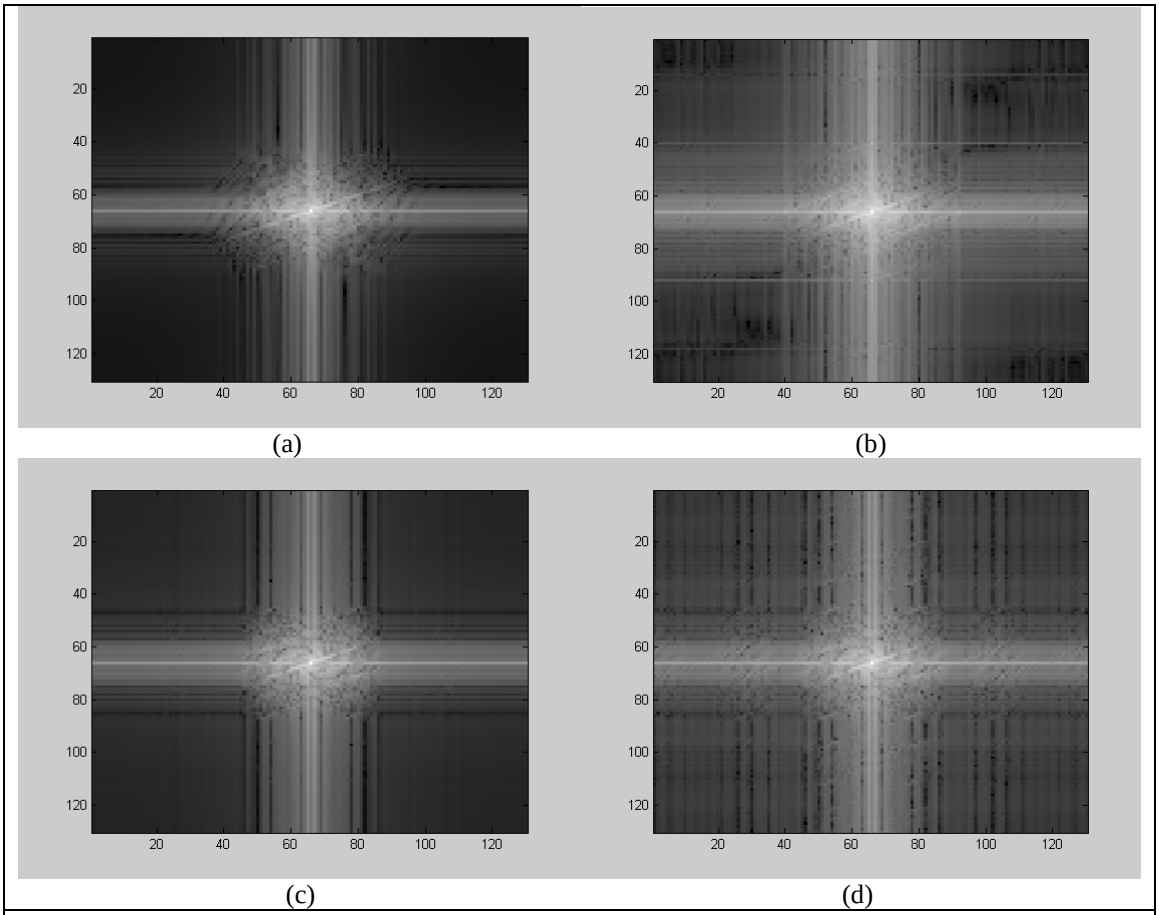


Figure 46 Full frequency spectra of the interpolated images. (a) Adaptive Gaussian Interpolation, (b) Gaussian radial basis function interpolation, (c) Bicubic Interpolation, (d) Bilinear Interpolation. Note the erroneous high frequency information present in (b) and (d).

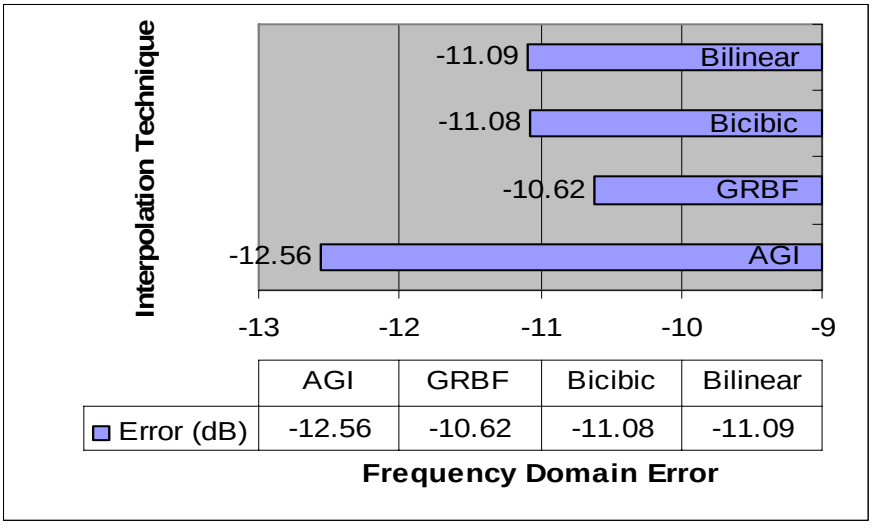
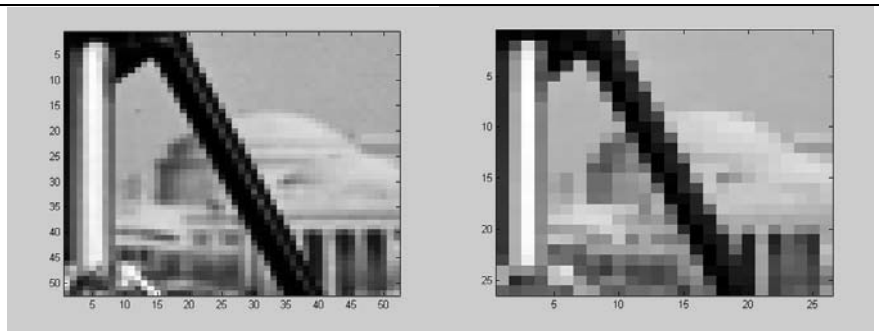


Figure 47 Mean square gray level error in frequency domain of the original image of the four interpolation techniques.

4.5 Edge detection performance

In determining how useful AGI might be for computer vision or target recognition, the results of AGI and the other techniques are subjected to standard edge detection algorithms. See Figures 48 and 49.



(a) (b)

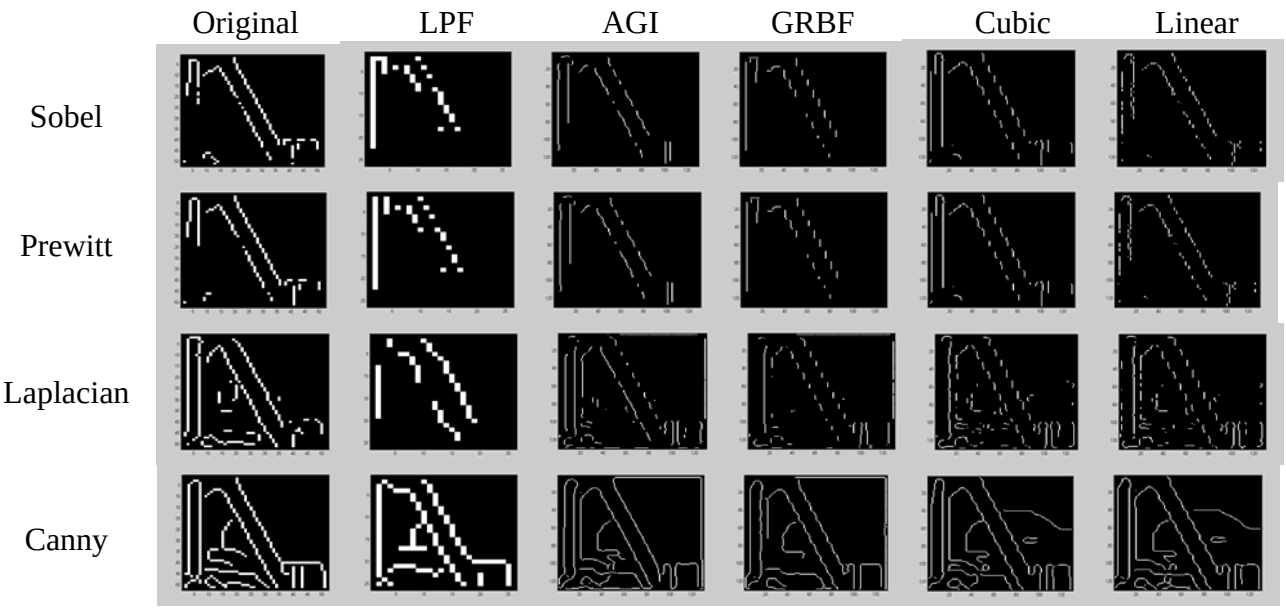
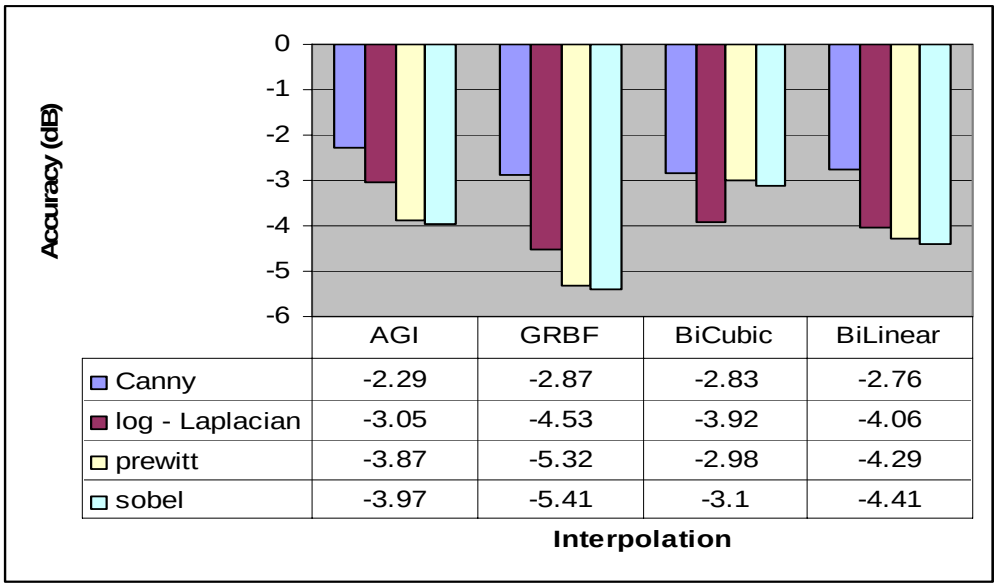


Figure 48 Comparison of edge-detection techniques for different interpolation methods. (a) is the original image and represents the first column of edge detection techniques, (b) is the down sampled version of (a) and represents the second column. Each column is labeled according to its respective interpolation method, and each row is labeled with the edge-detection technique.

To objectively quantify how well the interpolation techniques perform for edge detection, it is assumed that for any given edge detection algorithm more edges detected

for a given threshold indicates better results. Thus the objective analysis of edge detection results shown in Figure 53 displays the normalized sum total edge pixels detected.



	AGI	GRBF	BiCubic	BiLinear
Canny	-2.29	-2.87	-2.83	-2.76
log - Laplacian	-3.05	-4.53	-3.92	-4.06
prewitt	-3.87	-5.32	-2.98	-4.29
sobel	-3.97	-5.41	-3.1	-4.41

Figure 49. Evaluation of interpolation performance in edge detection. Here it is assumed that edge detection improves as interpolation improves for a given threshold. The amount of edge detection is the number of pixels with edges divided by the total number of pixels. According to this metric, interpolation techniques with larger amounts of edge detection are better techniques. The default threshold in MATLAB is used in all cases. Results are in dB using the amount of detection in the original image as the maximum detection (shorter bars indicate better results).

V. Conclusions and Recommendations

5.1 Conclusions

This thesis contributes a new and effective 2-D image interpolation technique that outperforms several other interpolation techniques. Overall, AGI yields significant improvements over other interpolation methods. Especially obvious is the decrease in the gray level mean square error of AGI over the other tested algorithms. Image frequency analysis also indicates that AGI yields improved accuracy in the frequency domain. Based on these results and the flexibility of AGI, it may be concluded that this algorithm has potential in a variety of applications. The best results were for the Lenna image, which supports use of AGI in the area of facial recognition.

5.2 Recommendations for Future Work

The research done here may be extended in numerous ways. The opportunity to make modifications to the technique developed here more effective or faster is present in several areas. Some possible modifications, significant alterations, and other directions in which the research might be advanced are considered below.

5.2.1 Modifications

The efficiency of AGI algorithm execution could be further improved. AGI has the potential to be highly parallel, and re-working the program code to take advantage of computer architecture could be beneficial in reducing execution time.

Other changes to AGI might include developing improved methods for calculating the parameters of the 2-D Gaussians, making the technique more adaptive by

allowing the total smoothness of each Gaussian to be adjusted for each pixel, and finding better ways to measure smoothness. A better way to determine the magnitude of the Gaussians could be developed to replace the time-consuming iterative technique used here.

More research might also be done in the area of the objective and subjective evaluation of results. Different ways to measure how well a given interpolation technique works could result in new ideas and improved output. Subjective evaluation can also be parameterized using surveys.

5.2.2 Significant Alterations

A significant change to the program might be made in the area of how the variables are used to calculate the covariance matrix. AGI uses a least squares technique to calculate parameters; however, it was found that this method does not make good use of the additional information taken from larger window sizes. Additional data available from larger windows should enable improved interpolation.

5.2.3 Other Advances

The AGI algorithm focuses on using a Gaussians to interpolate an image; however, further research could investigate more appropriate basis functions. Gabor functions and some wavelet functions have more flexibility than 2-D Gaussians and thus have the potential to be more adaptive. A central difficulty here is calculating the parameters needed to describe these functions using the data present in the image.

Adaptation to a third dimension is another potential area in which to continue this research. A 3-D Gaussian can represent three-dimensional images and could significantly contribute to the interpolation of 3-D data and to graphics design. Large efficiency improvements on the current AGI will be necessary to maintain reasonable processing times.

Bibliography

1. Burrus, C. S., R. A. Gopinath, and G.Haitao, *Introduction to Wavelets and Wavelet Transforms a Primer*, Prentice Hall, 1998

2. Duda, R. O., P. E. Hart, and D. G. Stork, *Pattern Classification,* Sec. Ed., Wiley 2001.

3. Eastman Kodak Company, 2003 http://www.kodak.com/global/en/digital/ccd/products/fullframe/fullframeFamilyMain.jhtml

4. Garcia-Leon, A. Probability and Random Processes for Electrical Engineering, Sec. Ed, Addison-Wesley, 1994.

5. Gustafson, S. C., V. E. Nikolaou, and F. Ahmed, "Image Smoothing with Minimal Distortion", Proc. SPIE, Vol. 2488, No. 02, Orlando FL, 1995.

6. Hahn, B. D., *Essential MATLAB® for Scientists and Engineers,* Sec. Ed., Oxford, 2002.

7. Lehmann, T. M., C. Gönner, and K. Spitzer, "Survey: Interpolation Methods in Medical Image Processing" IEEE Transactions on Medical Imaging, Vol. 18, No. 11, November 1999.

8. Li, X., and M. T. Orchard, "New Edge-Directed Interpolation," IEEE Transactions of Image Processing, Vol. 10, No. 10, October 2001.

9. Lim, J. S., *Two-Dimensional Signal and Image Processing*, Prentice Hall, 1990.

10. Meijering, E., "A Chronology of Interpolation: From Ancient Astronomy to Modern Signal and Image Processing", Proc. IEEE, Vol. 90, pp. 319-342, 2002.

11. Mitra, S. K., *Digital Signal Processing A Computer-Based Approach*, Sec. Ed., McGraw-Hill, 2001

12. Park, S. C., M. K. Park, and M. G. Kang, "Super-Resolution Image Reconstruction: A Technical Overview," *IEEE Signal Processing Magazine*, May 2003.

13. Rhea, W. J., "A New Technique for Enlargement and Reconstruction of Digital Sensor Imagery," *Remote Sensing*, Vol. 12, No. 3, pp 627-634, 1991.

14. Thévenaz, P., T. Blu, and M. Unser, "Interpolation Revisited," IEEE Transactions on Medical Imaging, Vol. 19, No. 7, July 2000.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
23-03-2004	**Master's Thesis**	Jun 2003 –Mar 2004

4. TITLE AND SUBTITLE

SUPER-RESOLUTION USING ADAPTIVE GAUSSIAN RADIAL BASIS FUNCTION INTERPOLATION

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Hunt, Terence, D., Second Lieutenant, USAF

5d. PROJECT NUMBER
If funded, enter ENR #

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GE/ENG/04-15

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
ANTHONY C. PRADIA, Lt Col, USAF
Chief, Comm & Info Support Div
AFTAC/LSC
1030 South Hwy A1A
Patrick AFB FL 32925
(321) 494-7225 DSN 854-7225
Anthony.Pradia@patrick.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Digital image interpolation using Gaussian radial basis functions has been implemented by several investigators, and promising results have been obtained; however, determining the basis function variance has been problematic. Here, adaptive Gaussian basis functions fit the mean vector and covariance matrix of a non-radial Gaussian function to each pixel and its neighbors, which enables edges and other image characteristics to be more effectively represented. The interpolation is constrained to reproduce the original image mean gray level, and the mean basis function variance is determined using the expected image smoothness for the increased resolution. Test outputs from the resulting Adaptive Gaussian Interpolation algorithm are presented and compared with classical interpolation techniques.

15. SUBJECT TERMS
Interpolation, Gaussian Process, Image Processing, Enhancement, Maximum Likelyhood Estimation, Numerical Analysis, Special Functions

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dr. Steven C. Gustafson, (ENG)
U	**U**	**U**	**UU**	58	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 785-3636, ext 4598: email: Steven.Gustafson@afit.edu

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18